UNIVERSIDADE FEDERAL DO CEARÁ

CAMPUS DE QUIXADÁ

CURSO DE ENGENHARIA DE SOFTWARE

RELATÓRIO – TRABALHO FINAL QUALIDADE DE SOFTWARE
Projeto TERMLT

Equipe:

Antônio Xavier de Sousa Neto

Professora:

Carla Ilane Moreira Bezerra

QUIXADÁ

Março, 2021

SUMÁRIO

1 DESCRIÇÃO DO PROJETO

O projeto selecionado é um projeto Java OO e Web de Código Aberto, é uma ferramenta de gerenciamento de terminologia compatível com SKOS baseada em tecnologias da Web Semântica. Ele permite gerenciar vocabulários consistindo de thesauri e ontologias. Ele também pode gerenciar documentos que usam termos dos vocabulários e analisar os documentos para encontrar ocorrências desses termos. O projeto pode ser encontrado abaixo.

Link do projeto: https://github.com/kbss-cvut/termit

Link do GitHub com os Arquivos:

Tabela 1 – Características do Projeto

Projeto	**LOC**	**# de classes**	**# de releases**
Termlt	**27.733**	**403**	**5**

2 AVALIAÇÃO DO PROJETO

2.1 Medição 1 – Antes de refatorar o projeto

Nessa Seção deve ser incluída a Tabela com a medição das métricas de coesão, acoplamento, complexidade, herança e tamanho, antes do projeto ser refatorado. Para isso será utilizada a ferramenta Understand. A Tabela 2 apresenta a descrição das métricas, faça uma tabela similar.

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

Sistema	**Coesão**	**Complexidade**				**Herança**			**Acoplamento**	**Tamanho**			
	LCOM2	ACC	SCC	EVG	MaxNet	DIT	NOC	IFANIN	CBO	LOC	CLOC	NIM	CDL
S1 antes da refatoração	**12355**	358	3725 **4705**	443	179	678	197 **1338**	513	**1651**	277173	7585	3253	403
S1 após refat. CS FE	**12377**	337	3585 **4530**	433	175	652	197 **1319**	470	**1638**				

S1 após refat. CS FE E RPB	**12390**	325	3539 **4448**	416	168	645	168 **1307**	513	1651 **1652**	27773	7585	3258	415
S1 após refat. CS FE E RPB e IC	**12476**	349	3610 **4572**	435	178	654	195 **1330**	481	**1676**	27.790	7585	3260	415
S1 após refat. CS FE E RPB e IC e DC	**11809**	334	3584 **4520**	428	174 **1351**	666	199	486	**1684**	27.790	7585	3268	420
S1 após refat. CS FE E RPB e IC e DC e SS													
	12565		**4530**		**1319**				**1638**	28.000	7585	3268	422

Tabela 3 – Métricas dos atributos internos de qualidade (MCCABE, 1976; CHIDAMBER; KEMERER, 1994; LORENZ; KIDD, 1994; DESTEFANIS *et al.*, 2014)

Atributos	Métricas	Descrição
Coesão	*Lack of Cohesion of Methods (LCOM2) (CHIDAMBER; KEMERER, 1994)*	Mede a coesão de uma classe. Quanto maior o valor dessa métrica, menos coesiva é a classe.
Acoplamento	*Coupling Between Objects (CBO) (CHIDAMBER; KEMERER, 1994)*	Número de classes que uma classe está acoplada Quanto maior o valor dessa métrica, maior é o acoplamento de classes e métodos.
Complexidade	*Average Cyclomatic Complexity (ACC) (MCCABE, 1976)*	Média da complexidade ciclomática de todos os métodos. Quanto maior o valor dessa métrica, mais complexa são a classes e métodos.
	Sum Cyclomatic Complexity (SCC) (MCCABE, 1976)	Somatório da complexidade ciclomática de todos os métodos. Quanto maior o valor dessa métrica, mais complexos são as classes e métodos.
	Nesting (MaxNest) (LORENZ; KIDD, 1994)	Nível máximo de aninhamento de construções de controle. Quanto maior o valor dessa métrica, maior é a complexidade de classes e métodos.
	Essential Complexity (EVG) (MCCABE, 1976)	Mede o grau na qual um módulo contém construtores não estruturados. Quanto maior o valor dessa métrica mais complexas são as classes e métodos.
Herança	*Number Of Children (NOC) (CHIDAMBER; KEMERER, 1994)*	Número de subclasses de uma classe. Quanto maior o valor dessa métrica maior é o grau de herança de un sistema.
	Depth of Inheritance Tree (DIT) (CHIDAMBER; KEMERER, 1994)	O número de níveis que uma subclasse herda de métodos e atributos de uma superclasse na árvore de herança. Quanto maior o valor dessa métrica maior é o grau de herança de um sistema.
	Bases Classes (IFANIN) (DESTEFANIS et al., 2014)	Número imediato de classes base. Quanto maior o valor dessa métrica, maior o grau de herança de um sistema.
Tamanho	*Lines of Code (LOC) (LORENZ; KIDD, 1994)*	Número de linhas de código, excluindo espaços e comentários. Quanto maior o valor dessa métrica, maior é o tamanho do sistema.
	Lines with Comments (CLOC) (LORENZ; KIDD, 1994)	Número de linhas com comentários. Quanto maior o valor dessa métrica maior o tamanho do sistema.
	Classes (CDL) (LORENZ; KIDD, 1994)	Número de classes. Quanto maior o valor , maior o tamanho do sistema.
	Instance Methods (NIM) (LORENZ; KIDD, 1994)	Número de métodos de instância. Quanto maior o valor dessa métrica maior é o tamanho do sistema.

2.2 Detecção dos Code Smells

Foram detectadas 308 instâncias de Code Smells, dentre eles estão: Shotgun Sugery, Refuse Parent Bequest, Intensive Coupling, God Class, Feature Envy, Dispersed Coupling e Data Class. Como são muitas instâncias e God Class e Data Class tem apenas 1 detecção. Optamos por escolher pelos outros 5 Smells que foram detectados, abaixo serão escolhidos 40 instâncias dessas 308.

Tabela 3 – Code smells do projeto.

Nome do Code Smell	**Quantidade**

Refused Parent Bequest	10
Feature Envy	10
Dispersed Coupling	8
Shotgun Sugery	10
Intensive Coupling	2

2.3 Medição 2 – Após Refatorar Code Smell: Feature Envy e Refused Parent Bequest

Os Code Smells Feature Envy e Refused Parent Bequest, foram removidos.do projeto, onde uma porcentagem de cada um foi removida, devido terem muitos deles, Foram Removidos 10 FE e 10 RPB, ficando assim 298 após a retaforação. Para a retirada do FE, foi utilizado a técnica de fatoração de Move Method e Extract Method, onde o Move Method foi utilizado em 8 dos 10 FE, e o Extract em 2.

Logo após para a retirada do RPB, foi utilizado as técnicas de Replace Inherance with Delegation, onde foi retirado as heranças que não era condizentes com as classes e também Extract Super class, onde foram criadas novas classes e delegado os métodos que faziam mais

sentido para elas e depois declarado a herança certa. Onde Replace Inherance foi usada mais vezes que a Extract super class. Depois da fatoração foi medida a qualidade interna, onde pode se notar em comparação a antes da fatoração uma melhora significativa nos atributos de herança e complexidade, onde antes Complexidade era 4695 e agora é 4448 e Herança antes da fatoração era 1388 e agora é 1307. Tivemos também um pouco de melhora em Coesão, e também em acoplamento.

Acredita-se que grande melhora e acoplamento se dá pelos métodos de fatoração e smells refatorados, onde foi movido muitos métodos pra onde eles fariam mais sentidos e removido e criado heranças também que faziam mais sentido.

2.4 Medição 3 – Após Refatorar Code Smell: Intensive Coupling

Na refatoração de Intensive Coupling. Como é um code smell que não existem métodos oficiais de refatoração, porém é um code smell que utiliza os métodos de sua classe excessivamente operações de outra, foi usada a estratégia de refatoração de Move method, onde foi criado um método dentro da classe onde chamava excessivamente esses métodos de outra classe.

2.5 Medição 3 – Após Refatorar Code Smell: Dispersed Coupling

O Dispersed Coupling também é um Code smell que não existem métodos oficiais de refatoração, então , como o Intensive , o Dispersed tem métodos da sua classe que acessam e dependem de muitas operações excessivas de outras classes, também foi usado o Move Method e Extract para criar um método na classe sem precisar chamar de outras ou Mover para a classe certa.

2.6 Medição 4 – Após Refatorar Code Smell: Shotgun Surgery

No caso do Shotgun, foi utilizado as técnicas de refatoração de Move Field e Move method, e alguns casos, alguns getters e setters que estavam na classes e não era utilizados foram removidos, assim removendo esses métodos que não eram utilizados.

2.7 Medição 5 – Após Refatorar Todos os Code Smells

No final, ao refatorar os Code Smells, abaixo na tabela mostrará o comparativo de como o código estava sem refatoração e depois para cada Code Smell.

Coesão	Complexidade	Herança	Acoplamento
12355	**4705**	**1338**	**1651**
12377↑	**4530↓**	**1319↓**	**1638↓**
12390↑	**4448↓**	**1307↓**	**1652↑**
12476↑	**4572↑**	**1330↑**	**1676↑**
11809↓	**4520↓**	**1351↑**	**1684↑**
12565↑	**4530↑**	**1319↓**	**1638↓**

Na tabela está o comparativo entre antes da fatoração e depois, e o que melhorou ou piorou dentro do código, o aumenta da coesão no código(seta pra cima), indica um aumento positivo, pois o código está mais coeso após as fatorações, como pode ver na Coluna 1 e 2, após isso a Coesão decai um pouco, e por ultimo sobre novamente, se for compararmos na tabela noacima, a piora da coesão se dá pelo fato após as refatorações dos Smells Dispersed e **Intensive Coupling, pelo fato de não haver muita experiência na suas refatorações. A mesma** coisa acontece com a Complexidade, ela diminui de forma positiva e depois aumenta na parte da Refatoração do Dispersed e Intensive Coupling, e o mesmo acontece com Herança e por fim acoplamento que no final diminui depois da refatorações de Shotgun Surgery.

Com isso, A coesão melhorou de 12355 para 12565, uma melhora de 2% Em complexidade, houve uma queda de 5%, então melhorou a complexidade do sistema de 4705 para 4530. Herança piorou muito pouco, menos de 2%, de 1338 a 1319 Acoplamento também teve uma piora de 1%, de 1651 para 1638 .

Dificuldades e Desafios

As maiores dificuldades apresentadas para fatoração do projeto, foram os code smells de Intensive e Dispersed Coupling, que nunca havia refatorado, e também o sistema em si, que era inglês e todo difícil de refatoração, por ser grande e etc.

3 COMPARAÇÃO DOS RESULTADOS

Leia o artigo: https://www.sciencedirect.com/science/article/pii/S0950584920301142?casa_token=xcwL1BwaRFUAAAAA:wZjXB0Wx-0FiMSpZSzyi0b7iRe7ZJOr8FdwihzEkvzeQHh0Iz6mxPCF769JgRiZ69TyfI5l8BP0

Faça uma comparação dos resultados do seu projeto de acordo com esse artigo.

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.

APÊNDICE A

Incluir possíveis documentos que possam ser gerados no desenvolvimento do sistema.