Knowledge Management and Analysis
Project 01: Bad smells

Armend Azizi
https://github.com/armendazizi1/proj1-bad-smell-detection

Section 1 - Ontology Creation

Structure of the created ontology

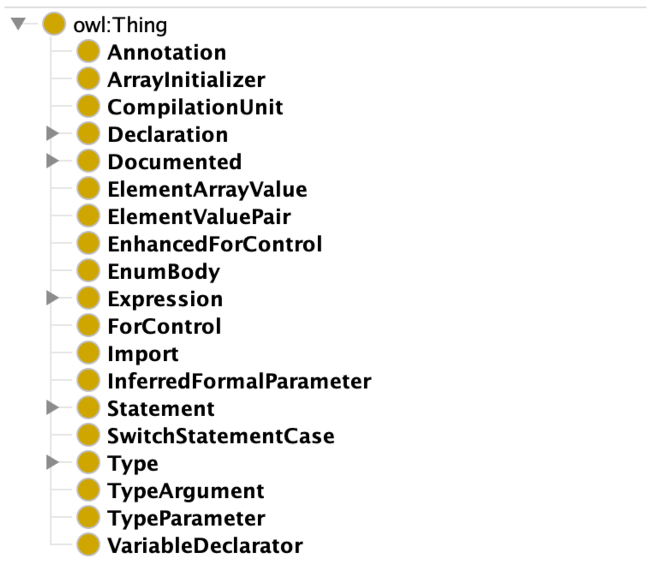

Figure 1: Highest level class hierarchy

In Fig. 1 we see the top-level classes from the class hierarchy defined in tree.py file that all inherit from class *Node* (e.g class Documented(Node):). In *onto-creator.py* file we have specified that when a node inherits from the *Node* class we classify it as subclass of class *Thing* (types.new_class(node.name, (Thing,))), and in Fig. 1 we see all the classes that inherit from Node classified correctly as subclasses of *Thing*.

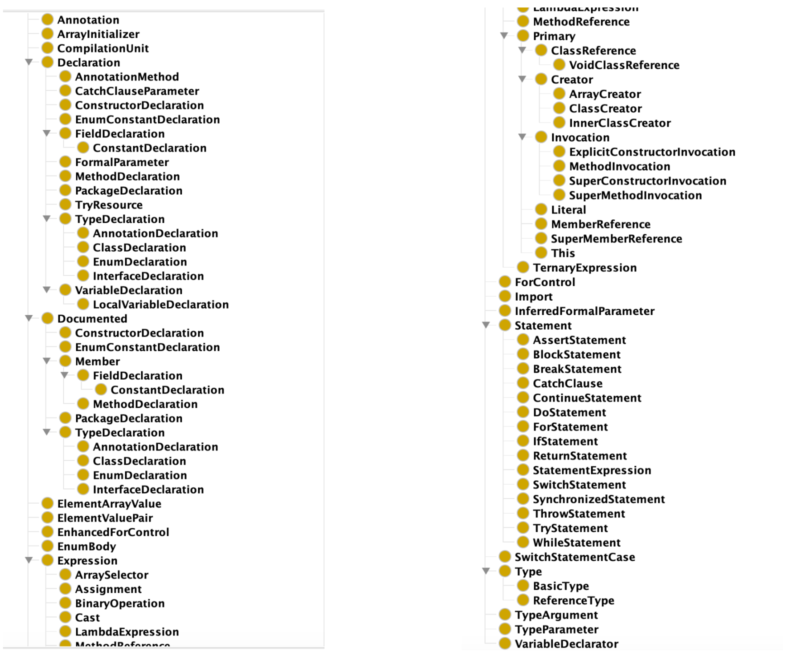

Figure 2: The expanded class hierarchy

Meanwhile, for the classes that do not inherit from class *Node* we specify their superclasses as they have specified it in tree.py, e.g *class ClassDeclaration(TypeDeclaration):* which means that class *ClassDeclaration* is a subclass of class *TypeDeclaration*, and in the left figure of Fig. 2 we see that *ClassDeclaration* is indeed classified correctly under *TypeDeclaration.* In code this is done with the command: *types.new_ class(node.name, (onto[x.id],))* in *onto-creator.py* file. Some classes have more than just one superclass from which they inherit (e.g *class TypeDeclaration(Declaration, Documented):*), and this behavior is also present in our owl class hierarchy, in the left image of Fig.2 we can see class *TypeDeclaration* both under the class *Declaration* and under the class *Documented*

Number of created classes and properties.

Report the count of created classes, and properties.

Type	Number
Class count	88
Object property count	2
Data property count	65

Table 1: Count of created classes and properties.

Section 2: Creation of ontology instances

Number of created instances.

Report the statistics (number of instances) for the individuals created for the considered project directory.

Class	Number of created individuals
ClassDeclaration	11
MethodDeclaration	152
FieldDeclaration	105
ConstructorDeclaratio	6
FormalParameter	165
AssertStatement	0
BlockStatement	143
BreakStatement	23
CatchClause	8
ContinueStatement	4
DoStatement	2
ForStatement	6
IfStatement	125
ReturnStatement	106
StatementExpression	446
SwitchStatement	8
SynchronizedStatemen	1
ThrowStatement	15
TryStatement	8
WhileStatement	10

Table 2: Number of created instances per class.

Section 3: Bad smell detection

Report the number of occurrences of each bad smell found in the code, as well as the list of code entities containing each smell for the considered project directory.

Bad Smell	Count
Long methods	10
Long constructors	0
Large classes	3
Methods with switch statements	8
Constructors with switch statements	0
Methods with long parameter list	4
Constructors with long parameter list	0
Data Classes	1

Table 3: Bad smells (total).

For each of the bad smells report a table with details of this type (if any instances exist):

Class Name	Method Name	Number of statements
GameControl	loadPGNMoves	97
ChessPuzzleProvider	insert	20
GameControl	requestMove	76
JNI	newGame	35
GameControl	getDate	26
PGNProvider	insert	31
GameControl	loadPGNHead	26
ChessPuzzleProvider	query	25
JNI	initFEN	88
JNI	initRandomFisher	87

Table 4: Long methods.

Class Name	Number of methods
Move	21
JNI	44
GameControl	63

Table 5: Large classes.

Class Name	Method Name	Number of switch statements
ChessPuzzleProvider	delete	1
PGNProvider	query	1
PGNProvider	getType	1
PGNProvider	update	1
ChessPuzzleProvider	getType	1
PGNProvider	delete	1
ChessPuzzleProvider	query	1
ChessPuzzleProvider	update	1

Table 6: Methods with switch statements.

Class Name	Method Name	Number of parameters
PGNProvider	query	5
GameControl	addPGNEntry	5
JNI	setCastlingsEPAnd50	6
ChessPuzzleProvider	query	5

Table 7: Methods with long list of parameters.

Class Name	Number of methods	Number of getter and setter methods
Valuation	1	1

Table 8: Data classes.