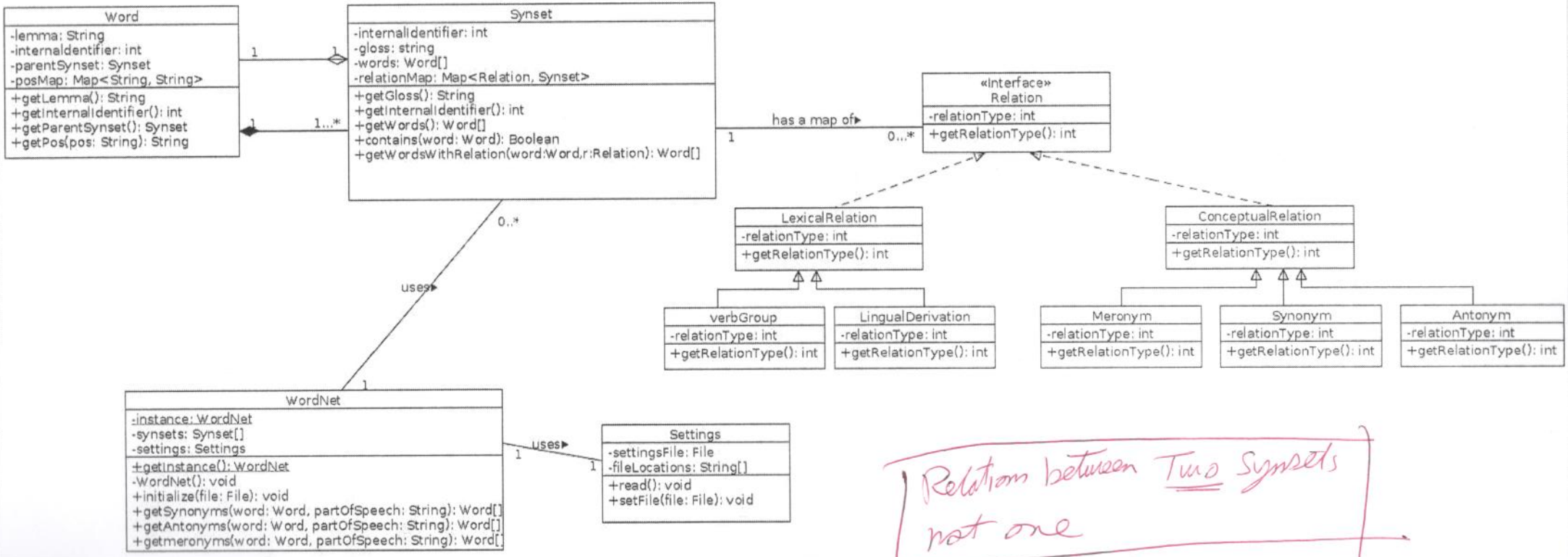
12/15

Word can have more than one POS
here you assume Word have one only
This is wrong and you need to fix
Mohamed Saad



Relation between Two Synsets
not one
Mohamed Saad

Q1, 4

4

Q2

```
class Synset {
    private int internalIdentifier;
    private String gloss;
    private Word[] words;
    private Map<Relation, Synset> replationMap;
    public Synset(Word[] words) {
    }
    public Word[] getWords() {
    }
    public boolean contains(Word word) {
    }
    public Word[] getWordsWithRelation(Word word, Relation relation) {
    }
}
```

==========

```
class Word{
    private String lemma;
    private int inernalIdentifier;
    private Synset parentSynset;
    private Map<String, String> posMap;
    public Word (Map<String, String> posMap){
    }
    public String getLemma(){
    }
    public Synset getParentSynset(){
```

```
    }
    public String getPos(String pos){
    }
}
==================================
interface Relation{
    public getrelationType();
}
==================================
class ConeptualRelation implements Relation{
    private relationType;
    public getRelationType(){
    }
}
==================================
class ConeptualRelation implements Relation{
    private relationType;
    public getRelationType(){
    }
}
==================================
 class LexicalRelation implements Relation{
    private relationType;
    public getRelationType(){
    }
```

```
}
====================================
class Synonym extends ConceptualRelation{
   private relationType;
   public getRelationType(){
   }
}
====================================
class Antonym extends ConceptualRelation{
   private relationType;
   public getRelationType(){
   }
}
====================================
class verbGroup extends LexicalRelation{
   private relationType;
   public getRelationType(){
   }
}
====================================
class WordNet {
   private static WordNet instance;
   private Synset[] synsets;
   private Settings settings;
   private WordNet() {
```

```
  }
  public static getInstance() {
  }
  public void initialize (File file) {
  }
  public Word[] getSynonyms ( Word word, String partOfSpeech ){
  }
  public Word[] getAntonyms(Word word, String partOfSpeech){\
  }
  public Word[] getMeronyms(Word word, String partOfSpeech){\
  }
}
==============================
class Settings {
        private File settingsfile;
        private String[] filelocations;
         public read() {
         }
          public setFile ( File file ){
           }
}
```

Q3)

Delegation Pattern, to extend and reuse the functionality of WordNet.

Q4)

1) Facade :- The WordNet Class is a Facade for other underlying wordNet classes which hides implementation details from client

2) singleton :- to ensure that only one instance of WordNet exists.

Q5)

The Properties File would look like an XML File.

```
<File Name>
    <path> File Path <path>
</File Name>
```

a- Public class DynamicLoading {

```
Public List<< class ? >> getClasses() {
    SettingsHolder holder = new settingsHolder();
    List <class<?>> classes = new . ——— —
List<string> Paths = holder.getPaths();

for(String Path : Paths) {
    make JarEntry entry

    String ClassName = getClassName(entry.getName)

    Class<?> cl = Class.forName(className);

    classes.add(cl);

}

}
```

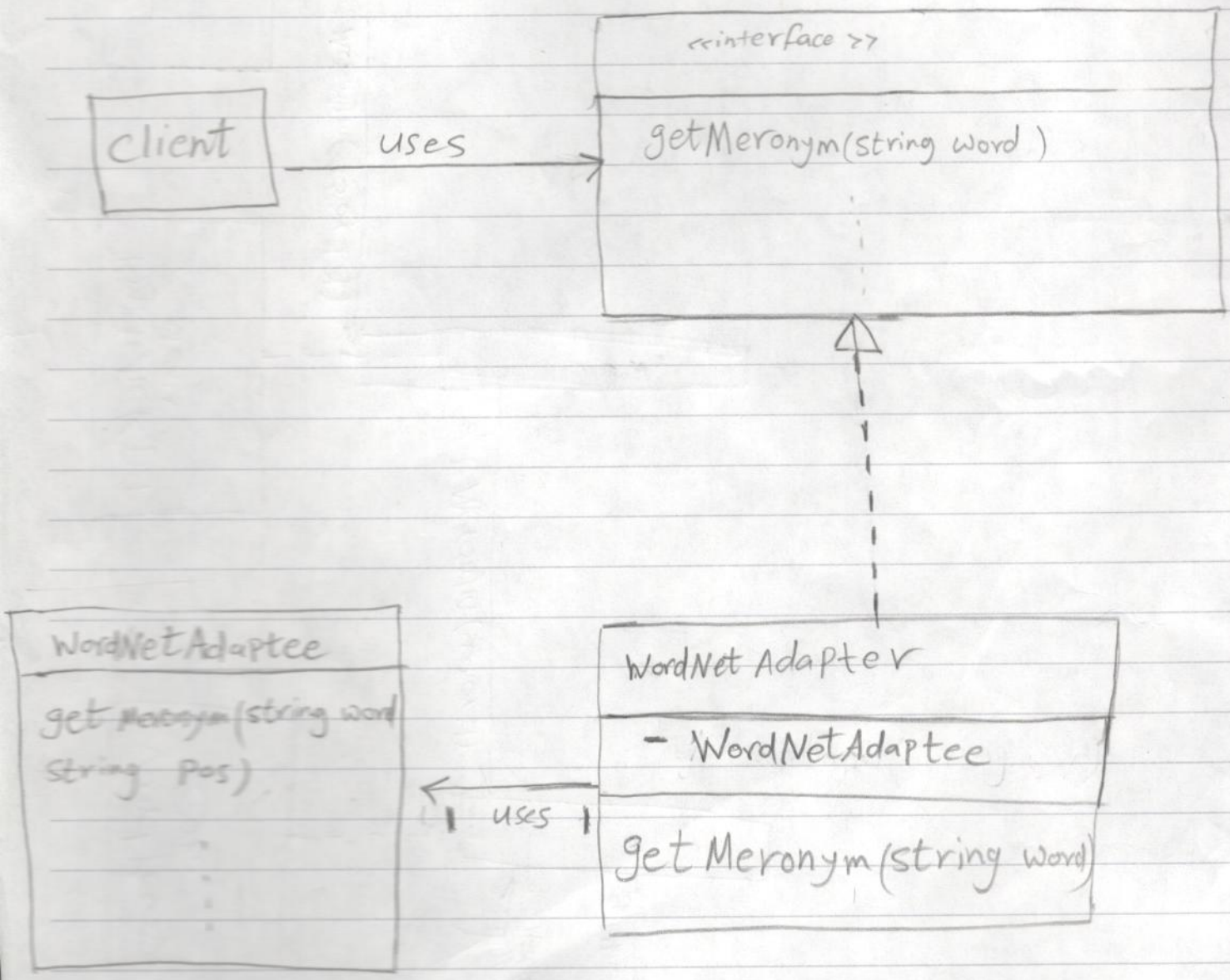
b-

```
Public Class RightClickPopMenu {
    WordProcessor P = WordProcessor.getInstance();
Public Void actionPerformed(ActionEvent e) {
    string word = e.getHighlightedWord();

    if(e.source == "synonyms")
        P.getSynonyms(word);

    else if (e.source == "meronyms")
        P.getmeronyms(word);

    else if (e.source == "antonyms")
        P.getAntonyms(word);
    }
}
```

Q6)

a) Adapter design Pattern

b)



(Q1)

① باهي مدحت

② عبد الرحمن يسري طه 37

③ علي محمد ثابت 39

④ محمد أحمد أبو الفتوح 50