Towards Variability-Aware Smells

POOR ANNOTATION TRACE

DEFINITION:

In the `Poor Annotation Trace` variability-aware smell scenario, source code annotations do not map quite precisely which code portion implements each feature.

EXAMPLE:

When a code element (method or class) holds annotation for more than one feature just in header of the command block (method or class) without any trace to which part of the block implements each annotated features in the header. Listing 7.1 shows a example of Poor Annotation Trace smell for the features *AStorage* (*${AStorage}*), *LLStorage* (*${LLStorage}*) and *Locking* (*${Locking}*) annotated at the top of the class *Stack* (lines 1–3), however, in the rest of the class *Stack*, there is no trace to which code portion of the class implements each one of the features *AStorage*, *LLStorage* and *Locking*.

Listing 7.1 Poor annotation trace example.

```
1  //#if ${AStorage} == "T"
2  //#if ${LLStorage} == "T"
3  //#if ${Locking} == "T"
4  class Stack <E> {
5   List<E> store = new ArrayList<E>();
6   List<E> store = new LinkedList<E>();
7   public void push(E e, Lock lock) {
8     lock.lock() ;
9     store.add(e);
10    lock.unlock();
11  }
12  E pop(Lock lock) {
13    lock.lock() ;
14    try { return store.remove(store.size()−1); }
15    finally { lock.unlock(); }
16  }
17 }
18 //#endif
19 //#endif
20 //#endif
```

<u>PROBLEM:</u>

`Poor Annotation Trace` smell compromises program comprehension, maintenance and evolution because it is very hard to identify an implementation for a specific feature when there is many feature implementations sharing the same source code in a class.