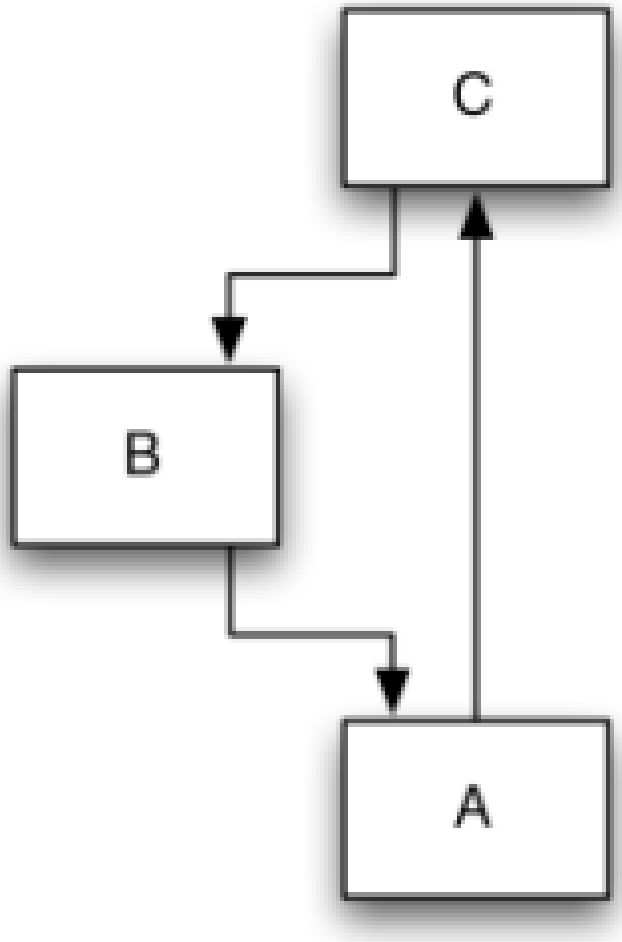
Identifying logical dependencies from co-changing classes

Stana Adelina Diana

Department of Computer and Information Technology
Politehnica University Timisoara, Romania

May, 2019

Dependencies



A `dependency` is a relationship that shows that an element, or set of elements, requires other elements for their specification or implementation. [UML Specification]

Figure 1: Dependencies in a project

Structural dependencies

Definition

Structural dependencies are the result of *source code analysis and can be extracted from : members, call parameters, local variables.*

```
class ClassA{
    ...
    public void doSomething(ClassB b){
        b.doWork();
    }
}
```

```
class ClassB{
    ...
    public void doWork(){
        ....
    }
}
```

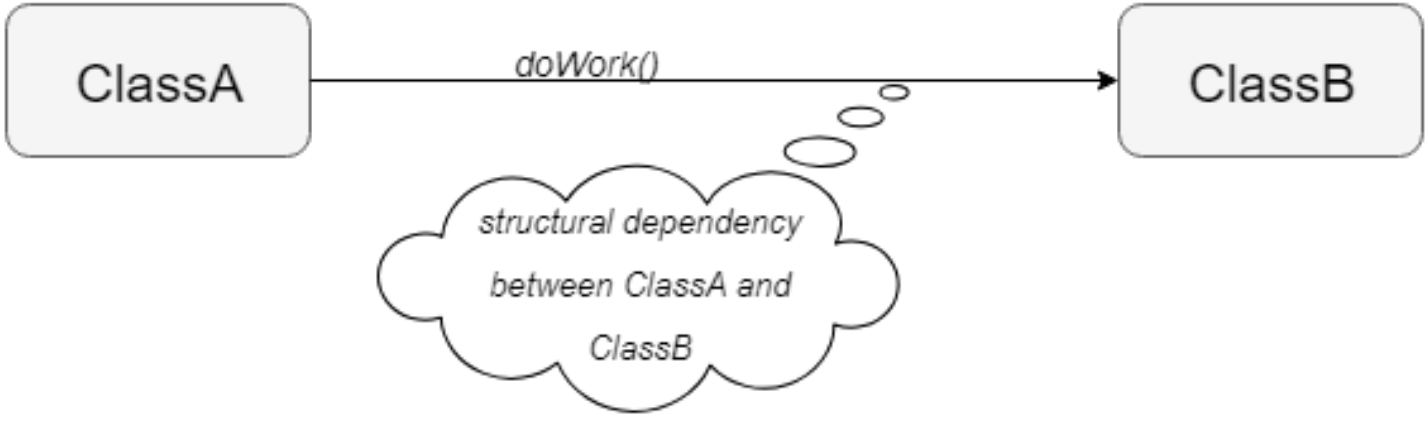


Figure 2: Example of structural dependency between two classes

Logical dependencies

Definition

Logical dependencies are the result of software history analysis and can reveal relationships that are not present in the source code code (structural dependencies).

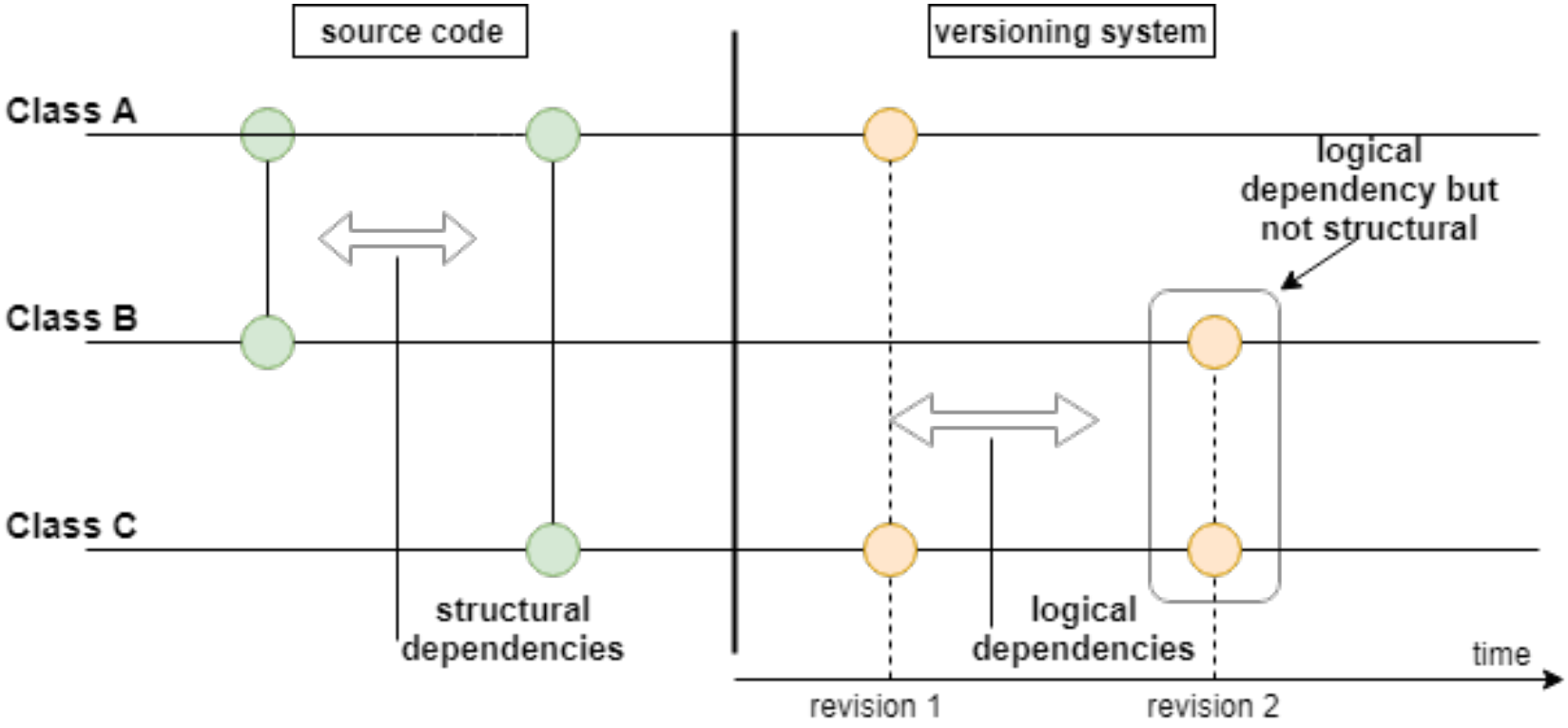


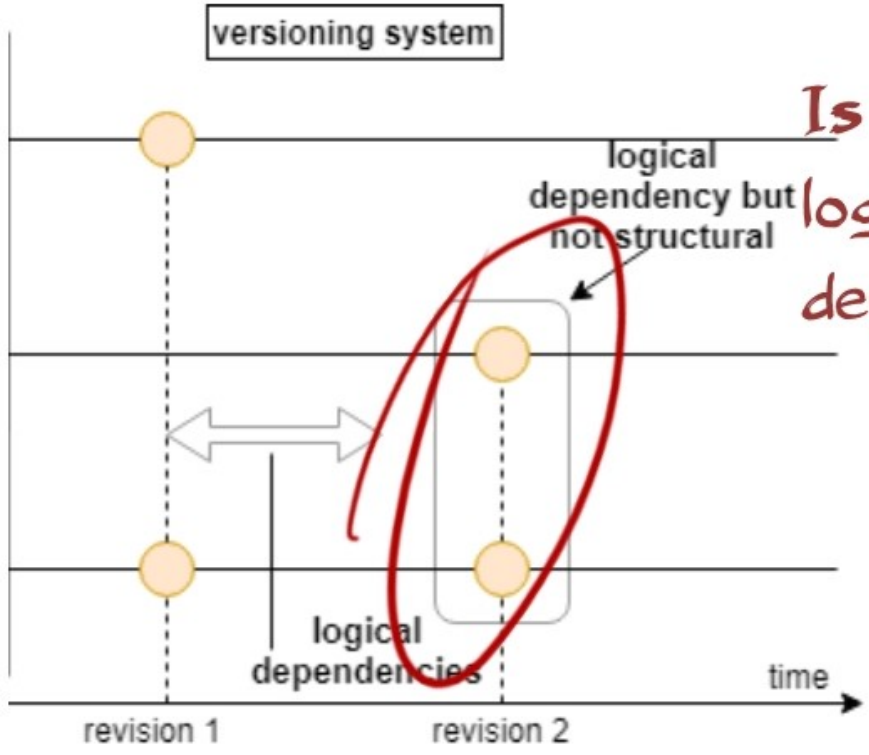
Figure 3: Example of logical and structural dependencies

Logical dependencies

Research questions

We build logical dependencies based on the following questions :
Question 1: *Which is the most frequent size for a commit transaction ?*
Question 2: *Is it necessary to set a threshold on the size of commit transactions which are considered to generate valid logical dependencies ?*
Question 3: *Considering changes which are only in comments as valid can lead to additional logical dependencies ?*
Question 4: *How many occurrences of a logical dependency are needed to consider it a valid logical dependency ?*
Question 5: *How does filtering affect the overlap between structural and logical dependencies ?*

Logical dependencies



Is this a logical dependency??

co-changes can happen from various reasons, not all software related

Co-changing classes

Biggest number of commits from our studied systems

Biggest number of commits from Github

10 000 commits

54 000 commits

304 000 pairs of co-changing classes

can generate aprox 1.5 million pairs of co-changing classes

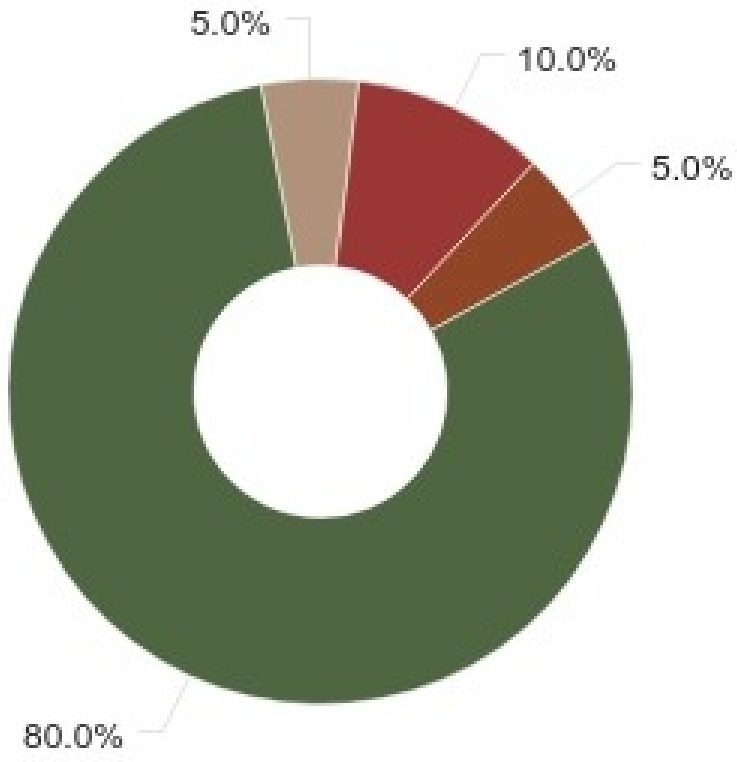
Filter co-changing classes, how?

commit size

code/comment change

number of co-change occurrence

Commit transaction size



cs <= 5	5.0%	5 < cs <= 10	10.0%
10 < cs <= 20	5.0%	cs > 20	80.0%

Conclusions

- Large number of structural dependencies are not doubled by logical ➔ systems partially stable
- + -3% for comments as a change
- The number of changed files taken into consideration influence the results
  - big threshold ➔ not so relevant logical dependencies
  - small threshold (5 10) ➔ more accurate results
- Filtering the logical dependencies after occurrences is good only for projects with a significant number of commits.

Future work

Investigate the cause for the large number of logical dependencies which are not overlapping with structural dependencies.