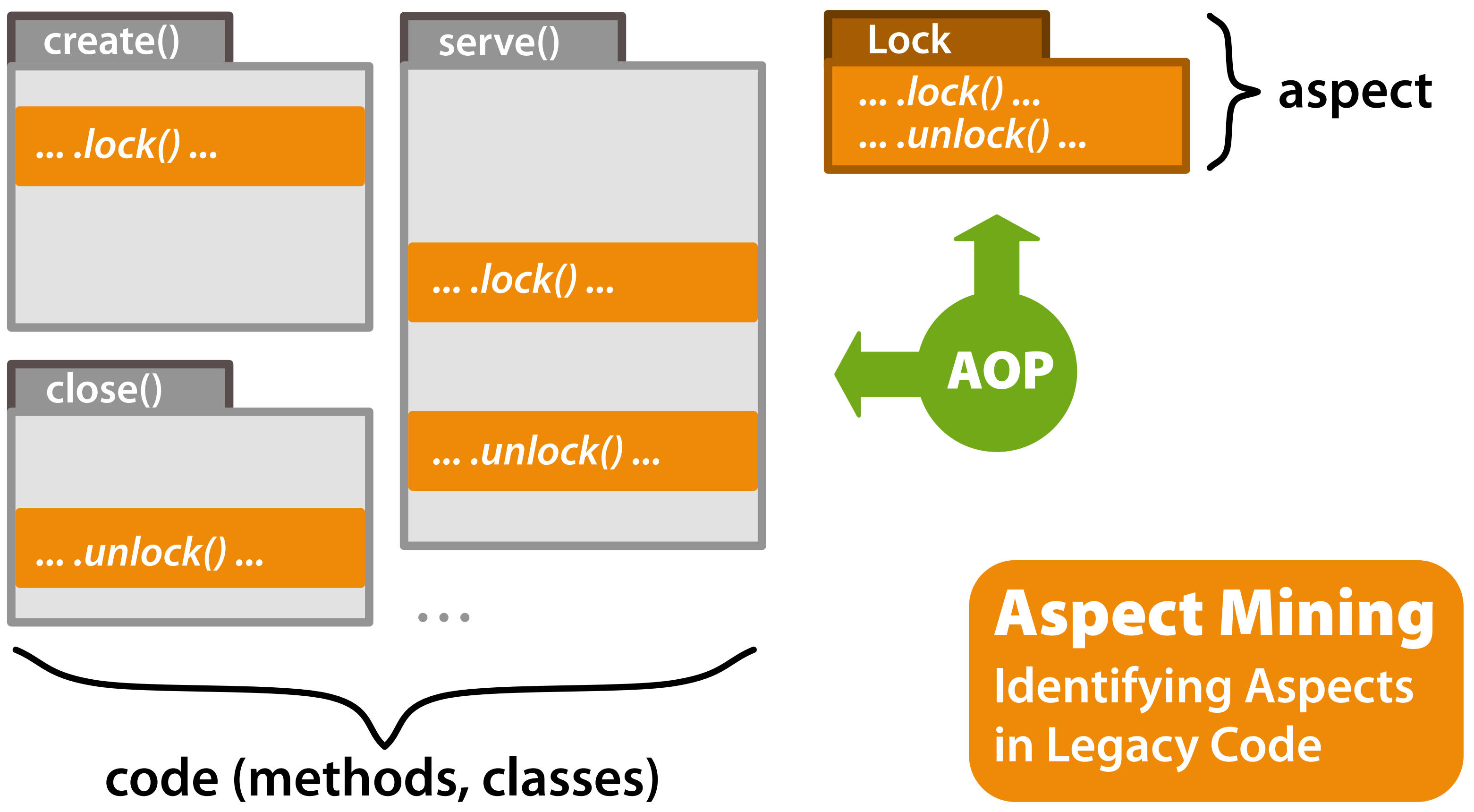
Aspect Mining for Large Systems

Silvia Breu
University of Cambridge
silvia@ieee.org

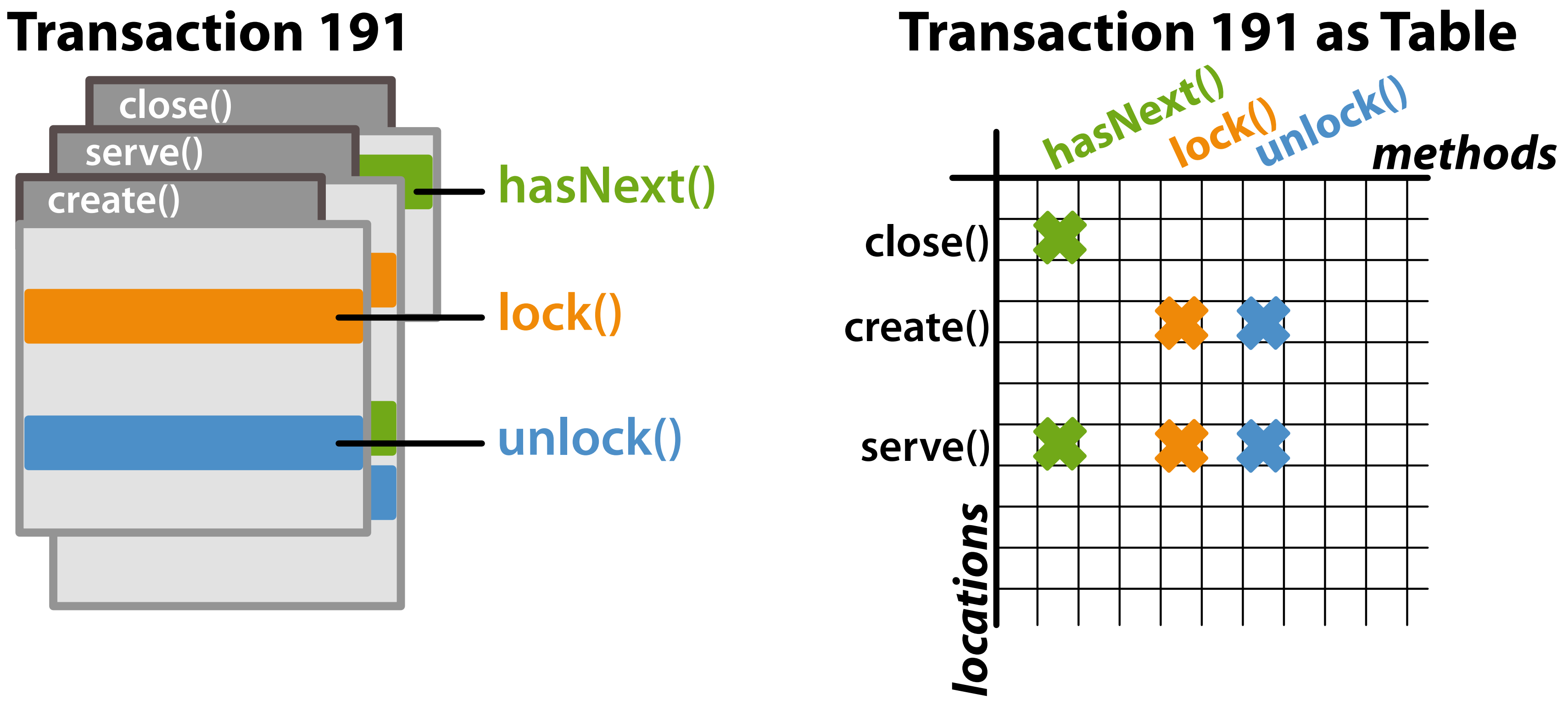
The Challenge



- ✦ large programs contain functionality that resists clean modularisation, also referred to as ***cross-cutting concern (CCC)***
- ✦ typical examples include *logging, debugging,* and *resource management (lock/unlock)*
- ✦ such scattered functionality is a weakness of current systems: it makes them hard to maintain, extend or change
- ✦ ***aspect-oriented programming (AOP)*** tries to remedy that by factoring this out into code-entities called ***aspects***
- ✦ for existing code to benefit from AOP, aspects have to be identified first, a task also called ***aspect mining***

The Idea – Step 1

Analysis of Version Archives

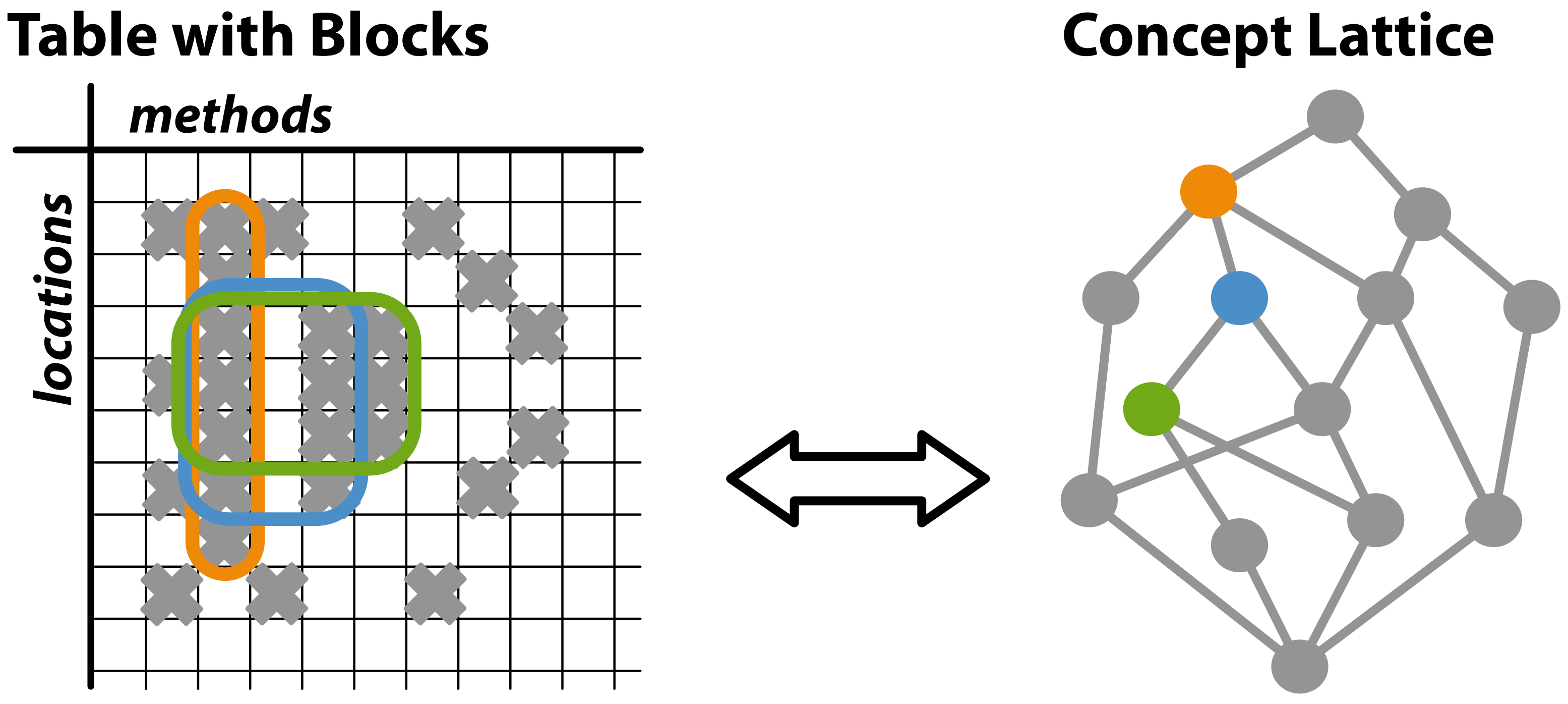


	hasNext()	lock()	unlock()
close()	✖		
create()		✖	✖
serve()	✖	✖	✖

- ✦ analyse CVS transactions for ***code additions***
- ✦ for each added method call in a code location make a cross ✖ in a table labelled with methods/locations

The Idea – Step 2

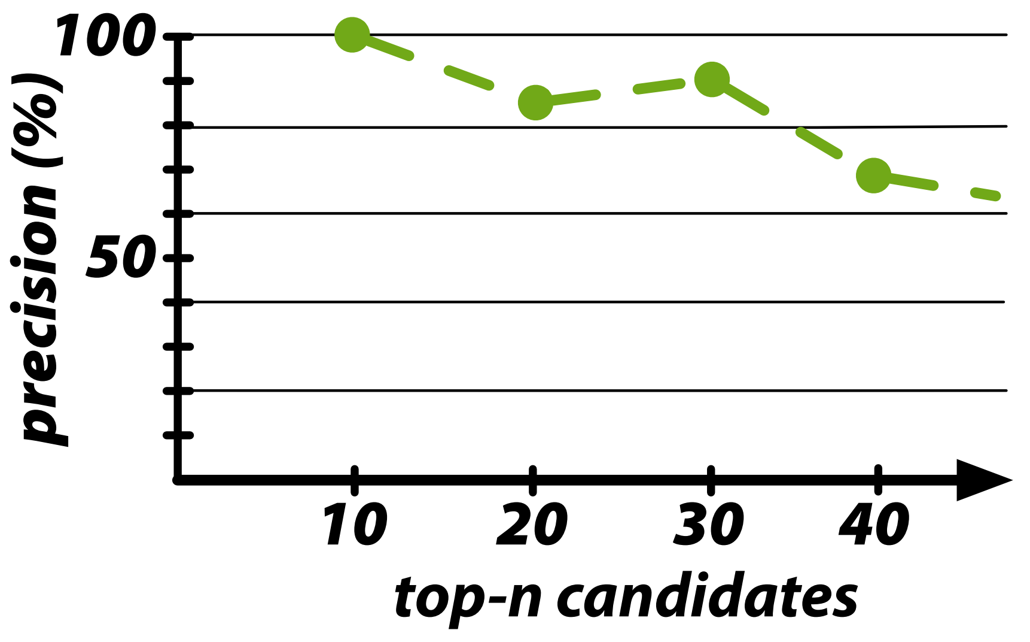
Formal Concept Analysis



- ✦ find ***maximal blocks*** in table as they ***represent CCCs*** or so-called ***aspect candidates***
- ✦ each block is also a concept in a lattice, which can be computed efficiently using ***formal concept analysis***

Results

Mining Results and Statistics (Eclipse 3.2M3)



	Eclipse 3.2M3
LOC	1.675.025
Transactions	97.900
Java files/methods	12.935/74.612

Aspect Candidates (≥8 locations)

methods	1	2	3	≥4
candidates	1878	362	88	24

- ✓ high ***precision***
- ✓ ***scalability*** to industrial-sized systems
- ✓ ***fast*** and ***efficient*** (on average 1 msec per transaction)

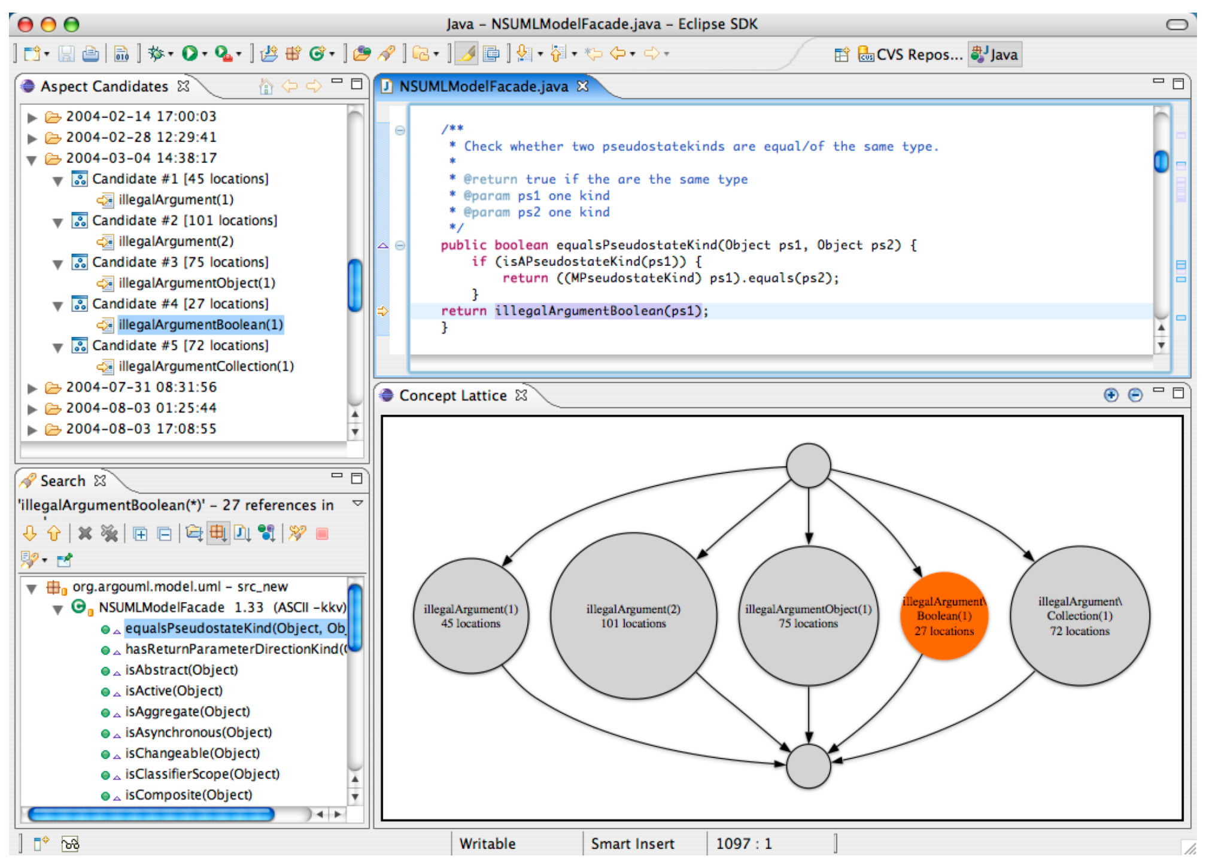
Status (1 year into PhD)

- ✦ Demo & Prototype ***"HAM" (History-Based Aspect Mining)***
- ✦ Publications

Mining Aspects from Version History. S. Breu, T. Zimmermann.
21st International Conference on Automated Software Engineering (ASE).

Mining Eclipse for Cross-Cutting Concerns. S. Breu, T. Zimmermann, and C. Lindig.
3rd International Workshop on Mining Software Repositories at ICSE (MSR).

What's next?



Aspect Candidates:

```
2004-02-14 17:00:03
2004-02-28 12:29:41
2004-03-04 14:38:17
  Candidate #1 [45 locations]
    illegalArgument(1)
  Candidate #2 [101 locations]
    illegalArgument(2)
  Candidate #3 [75 locations]
    illegalArgumentObject(1)
  Candidate #4 [27 locations]
    illegalArgumentBoolean(1)
  Candidate #5 [72 locations]
    illegalArgumentCollection(1)
2004-07-31 08:31:56
2004-08-03 01:25:44
2004-08-03 17:08:55
```

```java
/**
 * Check whether two pseudostatekinds are equal/of the same type.
 *
 * @return true if the are the same type
 * @param ps1 one kind
 * @param ps2 one kind
 */
public boolean equalsPseudostateKind(Object ps1, Object ps2) {
    if (isAPseudostateKind(ps1)) {
        return ((MPseudostateKind) ps1).equals(ps2);
    }
return illegalArgumentBoolean(ps1);
}
```

Concept Lattice: illegalArgument(1) 45 locations; illegalArgument(2) 101 locations; illegalArgumentObject(1) 75 locations; illegalArgument\Boolean(1) 27 locations; illegalArgument\Collection(1) 72 locations

Search: 'illegalArgumentBoolean(*)' – 27 references in

```
org.argouml.model.uml – src_new
  NSUMLModelFacade 1.33 (ASCII -kkv)
    equalsPseudostateKind(Object, Ob
    hasReturnParameterDirectionKind(
    isAbstract(Object)
    isActive(Object)
    isAggregate(Object)
    isAsynchronous(Object)
    isChangeable(Object)
    isClassifierScope(Object)
    isComposite(Object)
```

- ✦ ***Tool support***: build Eclipse plugin
- ✦ ***Deployment*** by programmers: is technique useful?
- ✦ Extension to find refactorings: ***include code deletions***

- ✦ ***Automatic tracking*** of (evolution of) CCCs: avoid re-mining after code changes
- ✦ ***Understanding concerns over time***: how do they evolve?

And What Do You Think?

Your opinion counts! Feel free to give your feedback below, or leave your email address and I'll get back to you.

Silvia will be here:
July 12