Program Comprehension as Fact Finding

Thomas D. LaToza, David Garlan,
James D. Herbsleb, Brad A. Myers

FSE 2007

Summary by Prof. Thomas LaToza

SWE 795, Spring 2017

Software Engineering Environments

Studies have long found large productivity differences

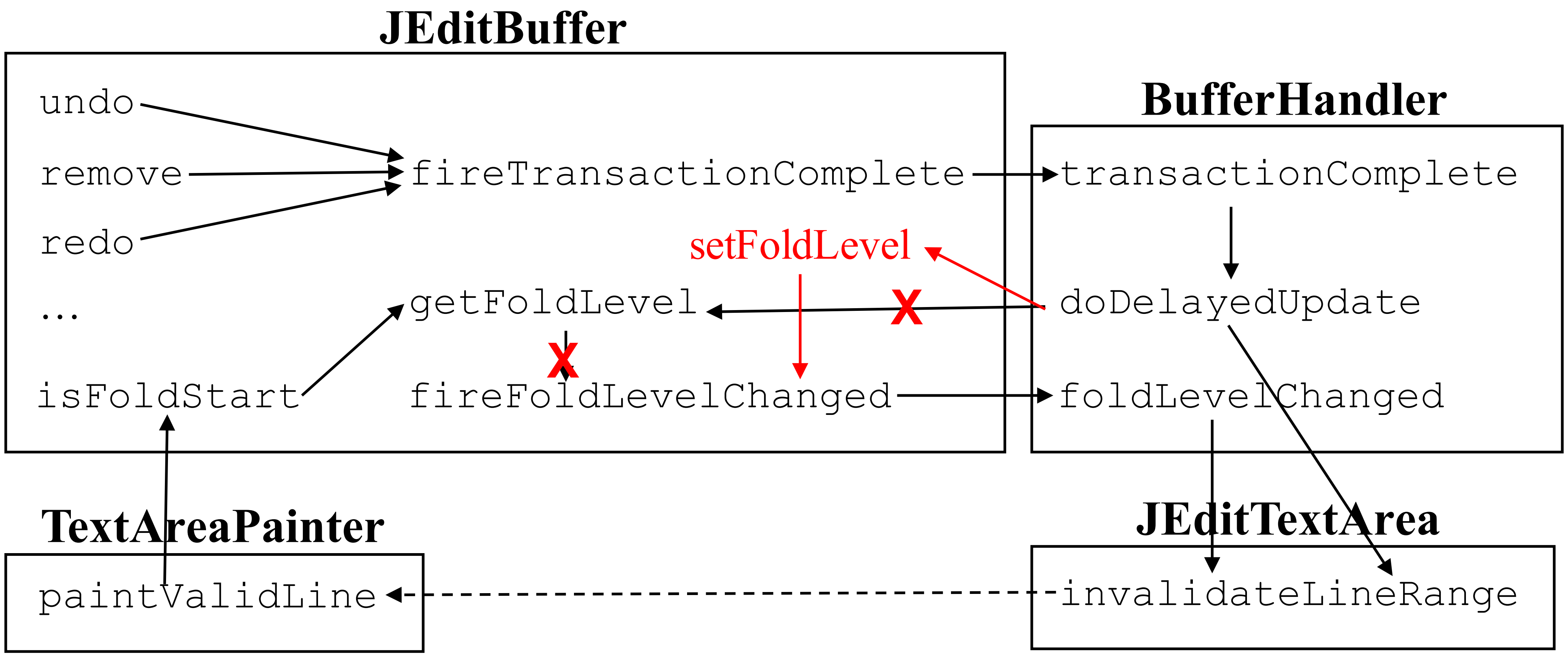
- Studies have long found **10x** differences in productivity (task time) between developers (e.g., [Sackman et al. 1968])
  - Why?
- Industry interest in hiring strong developers that are "10xers"
  - Who are such developers? Can it be taught?

Categorized developers into "novices" and "experts" based on expertise, compared how they worked

Participants grouped into "novices" and "experts"

	YRS industry experience	KLOC largest program	YRS Java experience	
	0	10	4	significant research programming
	0	7.5	3	
	0.5	1	few years	
	1.5	75	5	
10 "novices"	2	2	1	
	2.5	1	2	
	2.5	10	8	
	2.5	136	4	
	3	2	4	
	3	10	6	
	3	100	7	
3 "experts"	10	100	10	
	10.5	500	3	
Median	**2.5**	**10**	**4**	

Novice changes addressed the symptom of a design problem

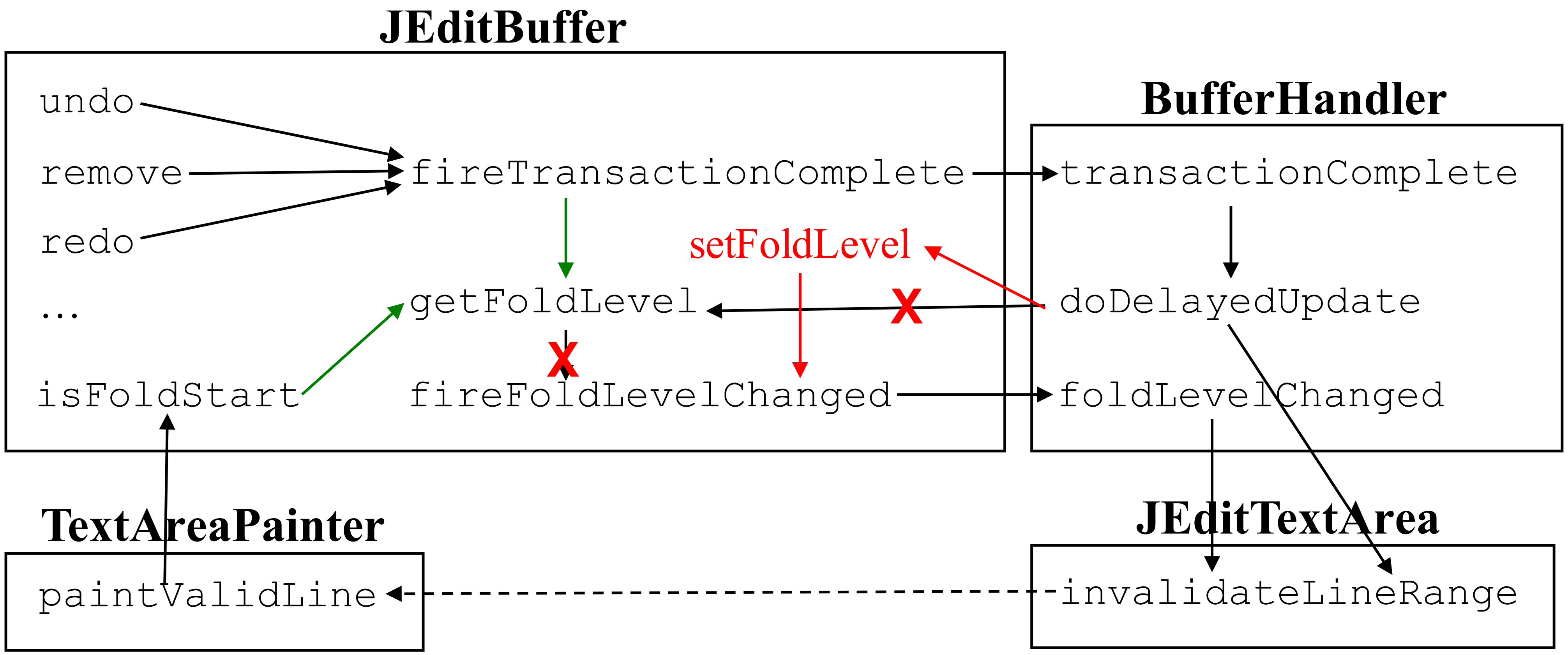


1 **Code smells**
Ignoring the return value of a getter
Using getter for its effects

1 novice change	Added debug statement
1 novice change	Removed call to getFoldLevel
8 novice + 1 expert changes	BufferHandler triggers refresh by a setter

2 **“Architecturally questionable”**
Changing buffer state from another component

Most expert changes addressed the cause of a design problem



1 **Code smells**

Ignoring the return value of a getter

Using getter for its effects

1 novice change	Added debug statement
1 novice change	Removed call to getFoldLevel
8 novice + 1 expert changes	BufferHandler triggers refresh by a setter

2 **“Architecturally questionable”**

Changing buffer state from another component

2 expert changes	Moved call from BufferHandler to JEditBuffer

Excerpts learned facts at a higher level of abstraction

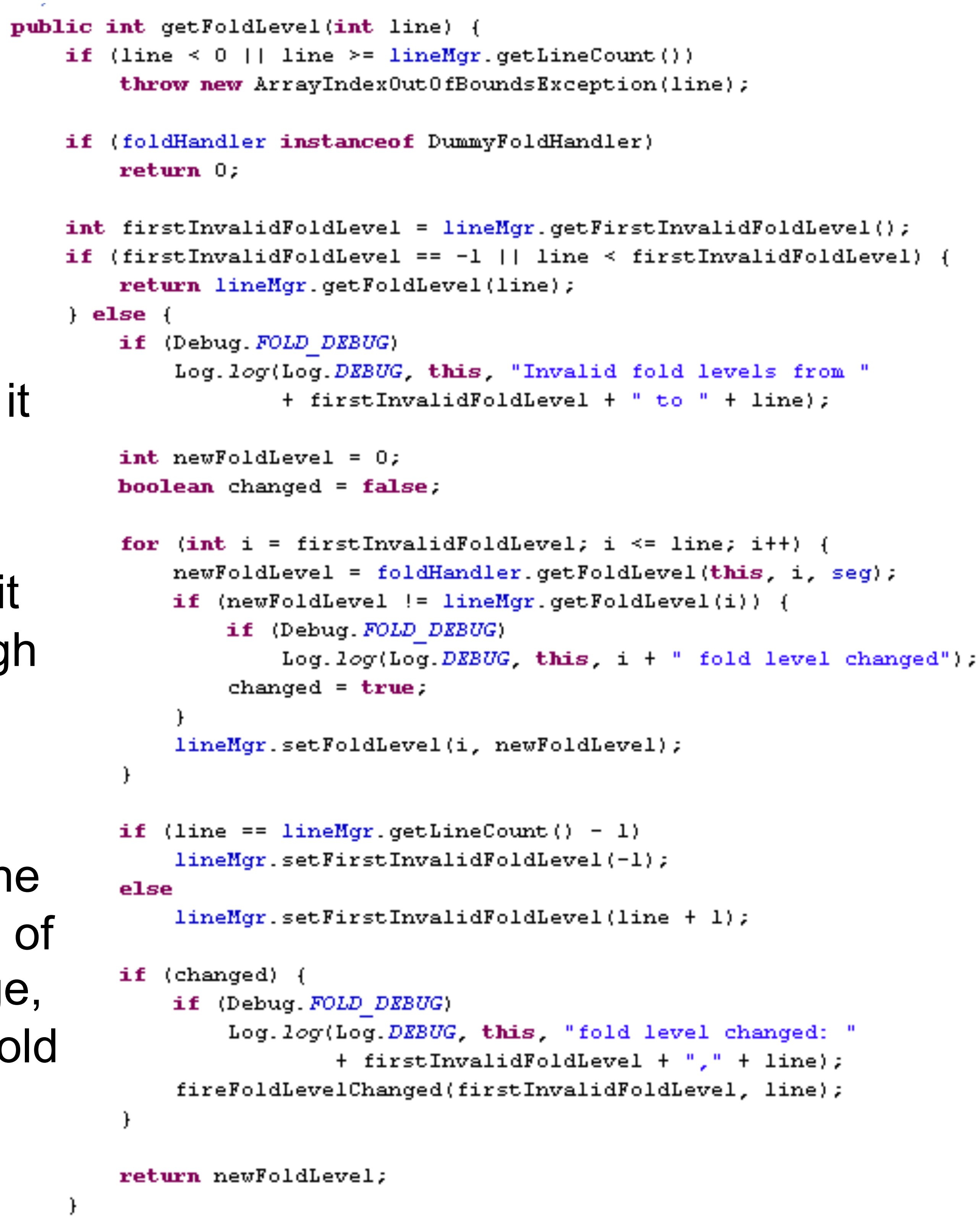
EXPERTS

“Well, this is just updating a cache” *(1 min)*

NOVICES

“What it did was it…computes the new line number and fires an event. But I didn’t see it change any state.” *(38 mins, 10 mins reading getFoldLevel)*

“So what it does, it starts off from this line, it has this firstInvalidFoldLevel, it goes through all these lines, it checks whether this fold information is correct or not, which is this newFoldLevel, this is supposed to be the correct fold level. If that is not the case in the data structure, it needs to change the state of the buffer. It creates this, it does this change, it sets the fold level of that line to the new fold level.” *(51 mins, 12 mins reading getFoldLevel)*



```
public int getFoldLevel(int line) {
    if (line < 0 || line >= lineMgr.getLineCount())
        throw new ArrayIndexOutOfBoundsException(line);

    if (foldHandler instanceof DummyFoldHandler)
        return 0;

    int firstInvalidFoldLevel = lineMgr.getFirstInvalidFoldLevel();
    if (firstInvalidFoldLevel == -1 || line < firstInvalidFoldLevel) {
        return lineMgr.getFoldLevel(line);
    } else {
        if (Debug.FOLD_DEBUG)
            Log.log(Log.DEBUG, this, "Invalid fold levels from "
                    + firstInvalidFoldLevel + " to " + line);

        int newFoldLevel = 0;
        boolean changed = false;

        for (int i = firstInvalidFoldLevel; i <= line; i++) {
            newFoldLevel = foldHandler.getFoldLevel(this, i, seg);
            if (newFoldLevel != lineMgr.getFoldLevel(i)) {
                if (Debug.FOLD_DEBUG)
                    Log.log(Log.DEBUG, this, i + " fold level changed");
                changed = true;
            }
            lineMgr.setFoldLevel(i, newFoldLevel);
        }

        if (line == lineMgr.getLineCount() - 1)
            lineMgr.setFirstInvalidFoldLevel(-1);
        else
            lineMgr.setFirstInvalidFoldLevel(line + 1);

        if (changed) {
            if (Debug.FOLD_DEBUG)
                Log.log(Log.DEBUG, this, "fold level changed: "
                        + firstInvalidFoldLevel + "," + line);
            fireFoldLevelChanged(firstInvalidFoldLevel, line);
        }

        return newFoldLevel;
    }
```

Experts explained more, helping them better respect constraints & reason about implications

EXPERT

“What's going on is that when you're inserting text you could actually be doing something that makes the folds status wrong. … If fox is under brown and I'm right at fox and I hit backspace. Then I would need to update my fold display to reflect the new reality, which is that it's in a different place. “

3 NOVICES + 1 Expert

Gave up moving update after

- Explaining why call was there
- Explaining purpose of BufferHandler
- Bug

Rejected task’s critique from a false fact (Expert)

Questions for discussion

- Overall reaction to the paper
- During what activities is this model applicable?
  - Could it be used to describe debugging behavior?
- How does this model relate to information needs models?

- What aspects of the program comprehension process are not addressed by this model?