



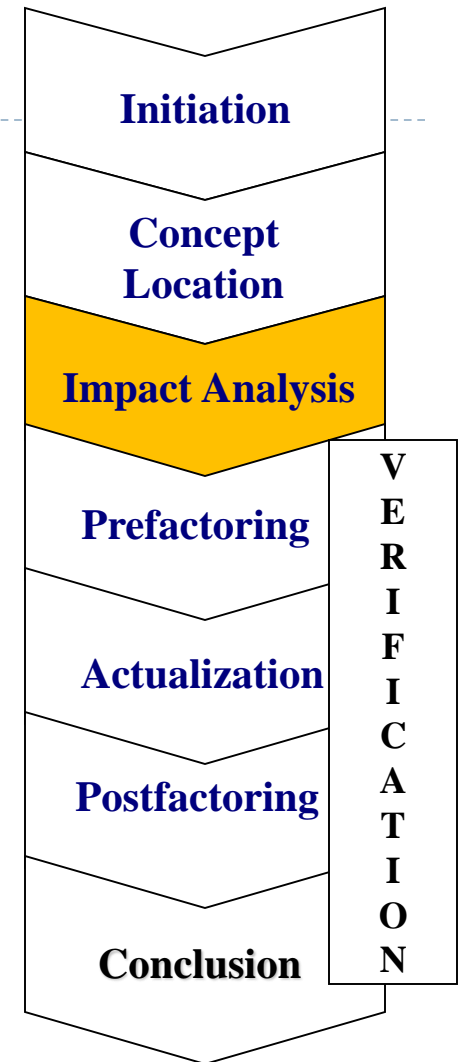
Impact Analysis



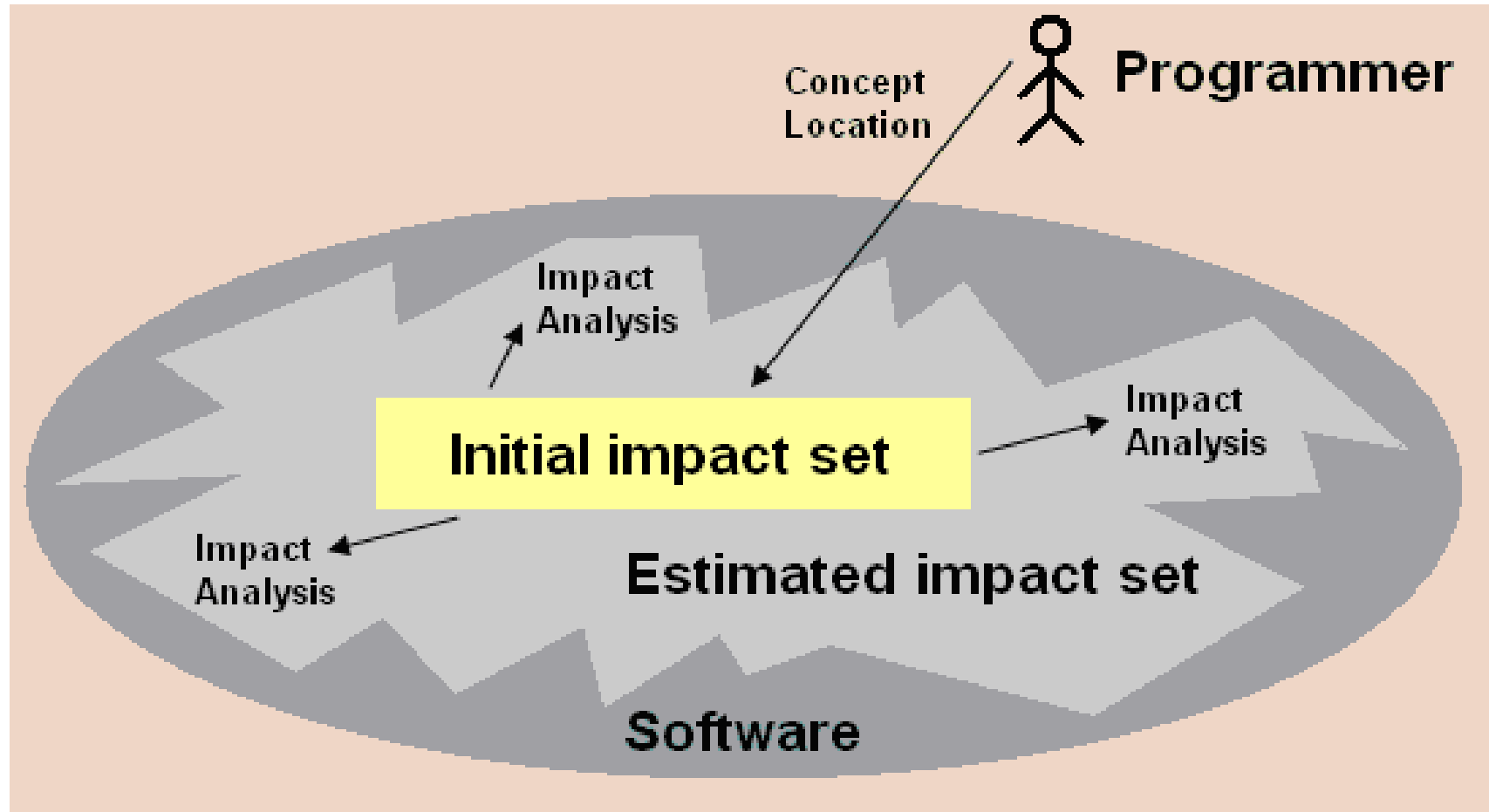
Roberto A. Bittencourt
Based on Rajlich's slides

7 Impact analysis (IA)

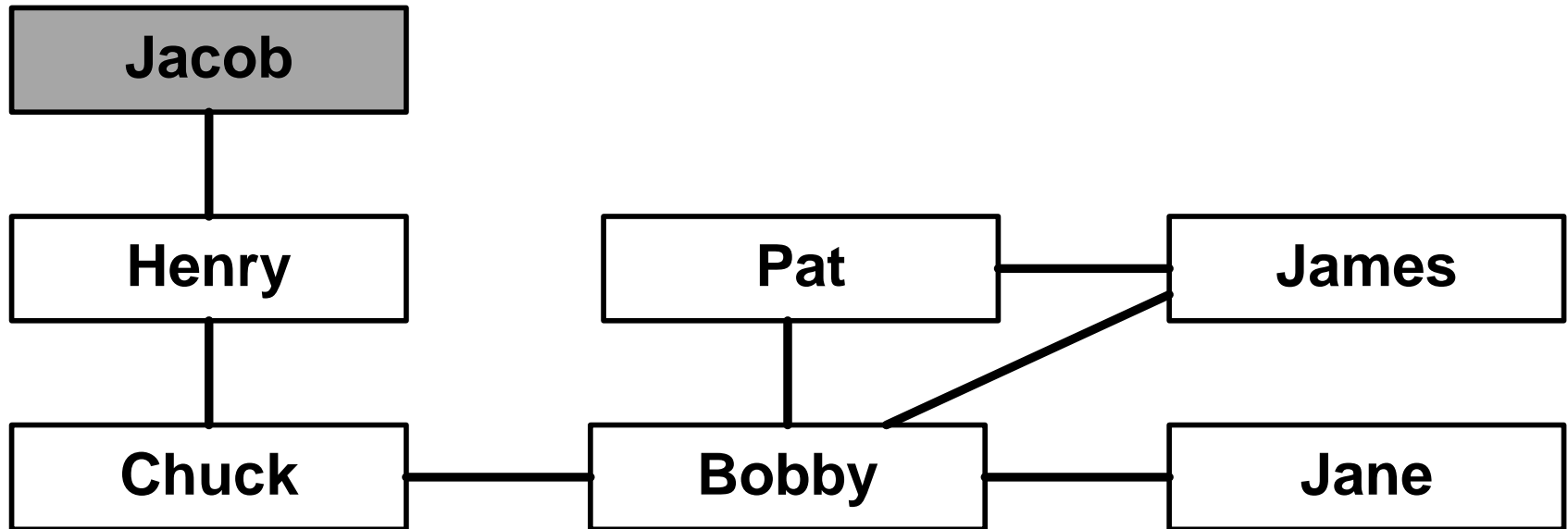
- ▶ Determines the strategy and impact of change
- ▶ Classes identified in concept location make up the *initial impact set*
- ▶ Class dependencies are analyzed, and impacted classes are added to the impact set
- ▶ Produces *estimated impact set*



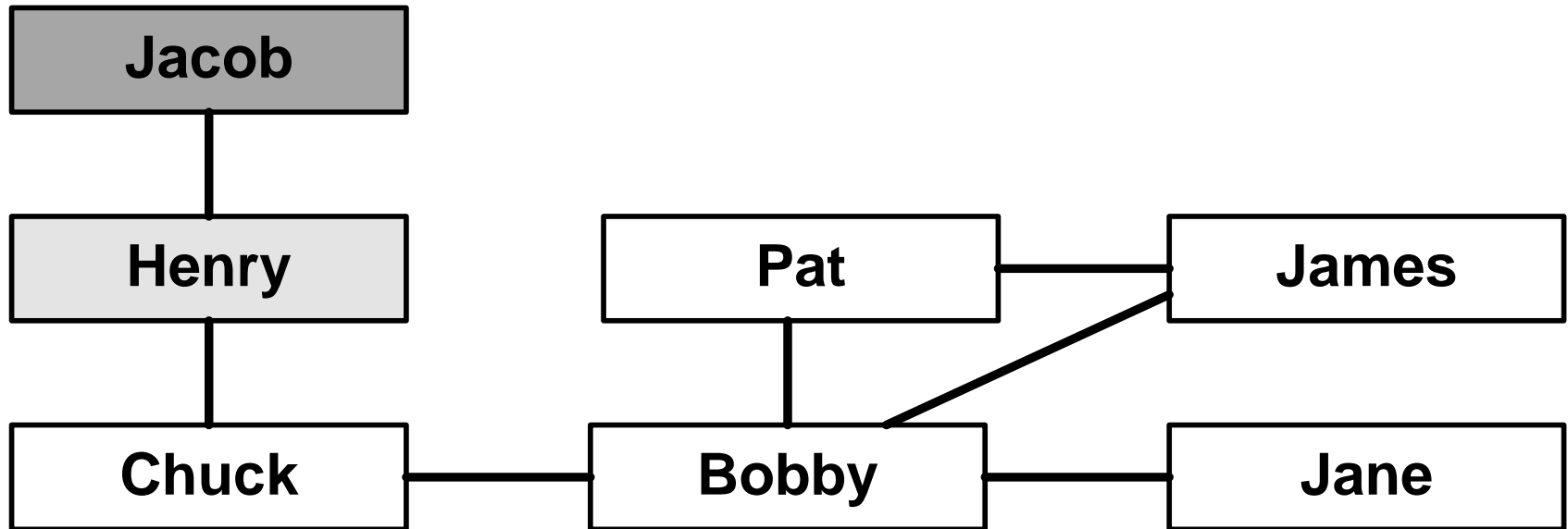
Initial and estimated impact set



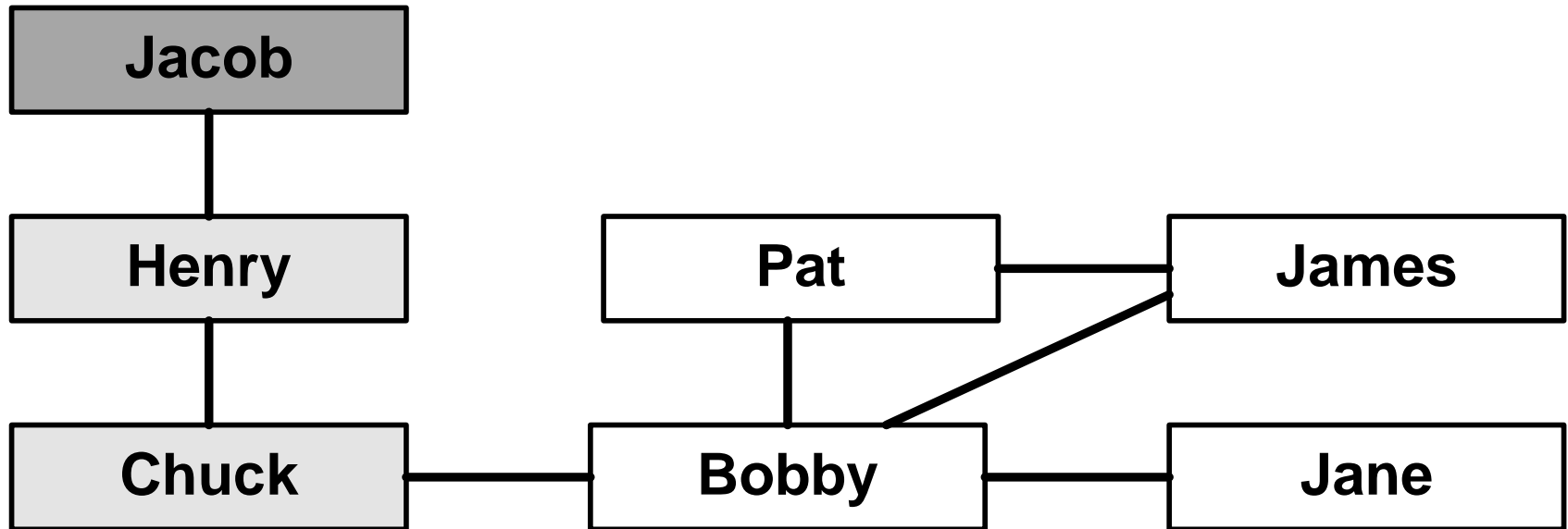
Example personal schedules



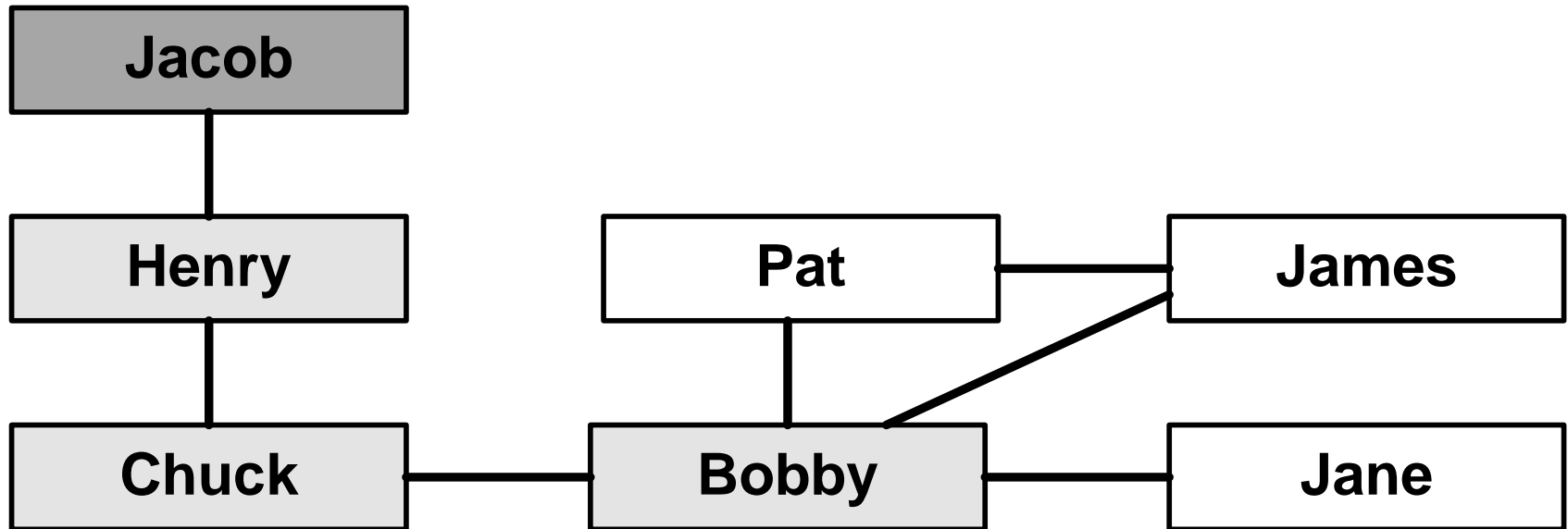
Example personal schedules



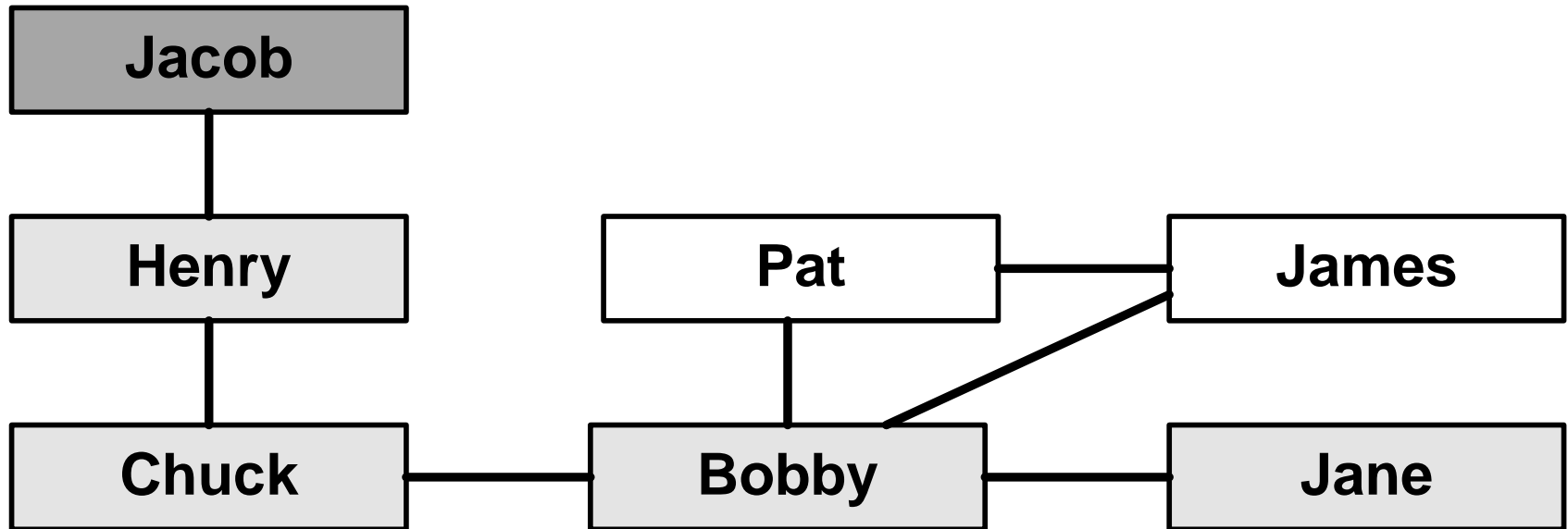
Example personal schedules



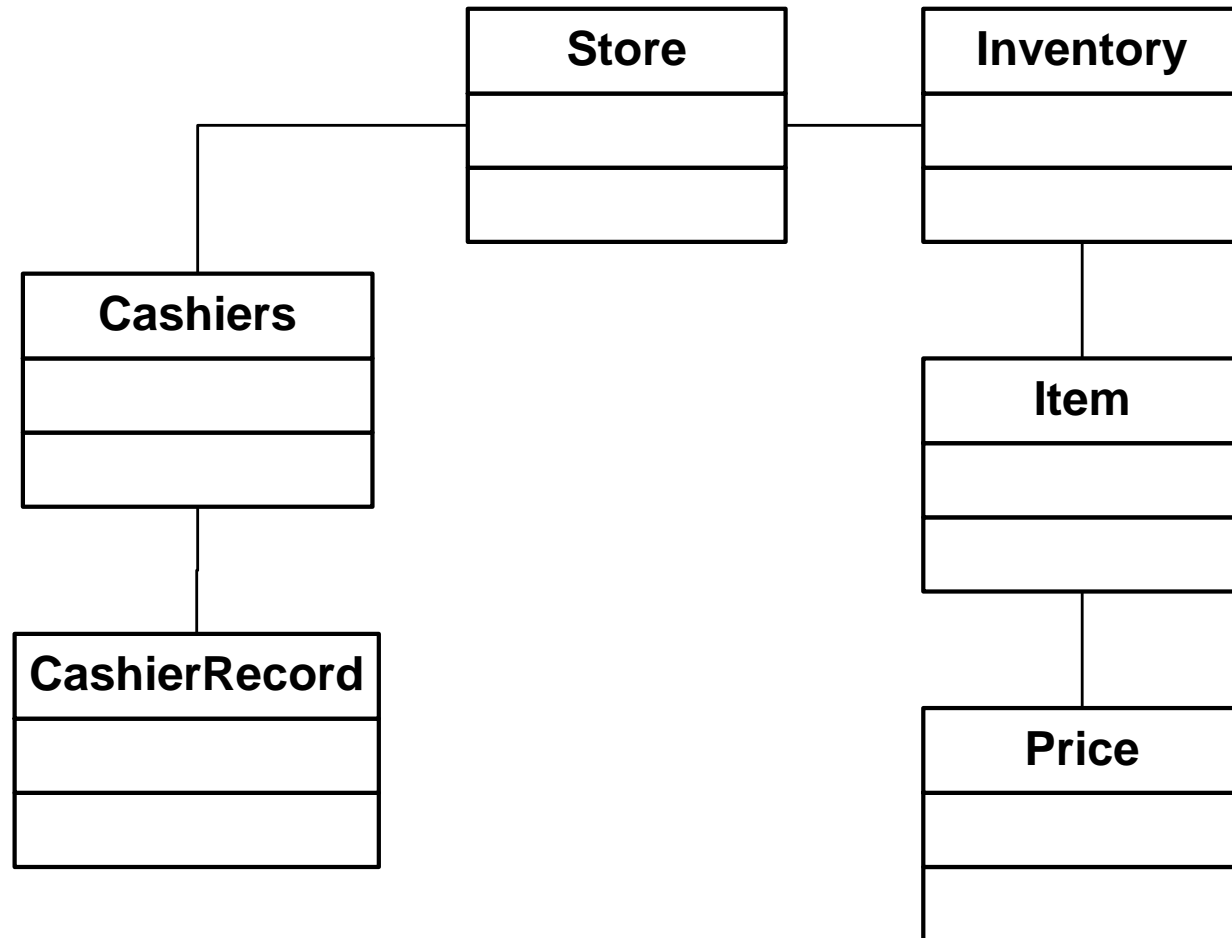
Example personal schedules



Example personal schedules



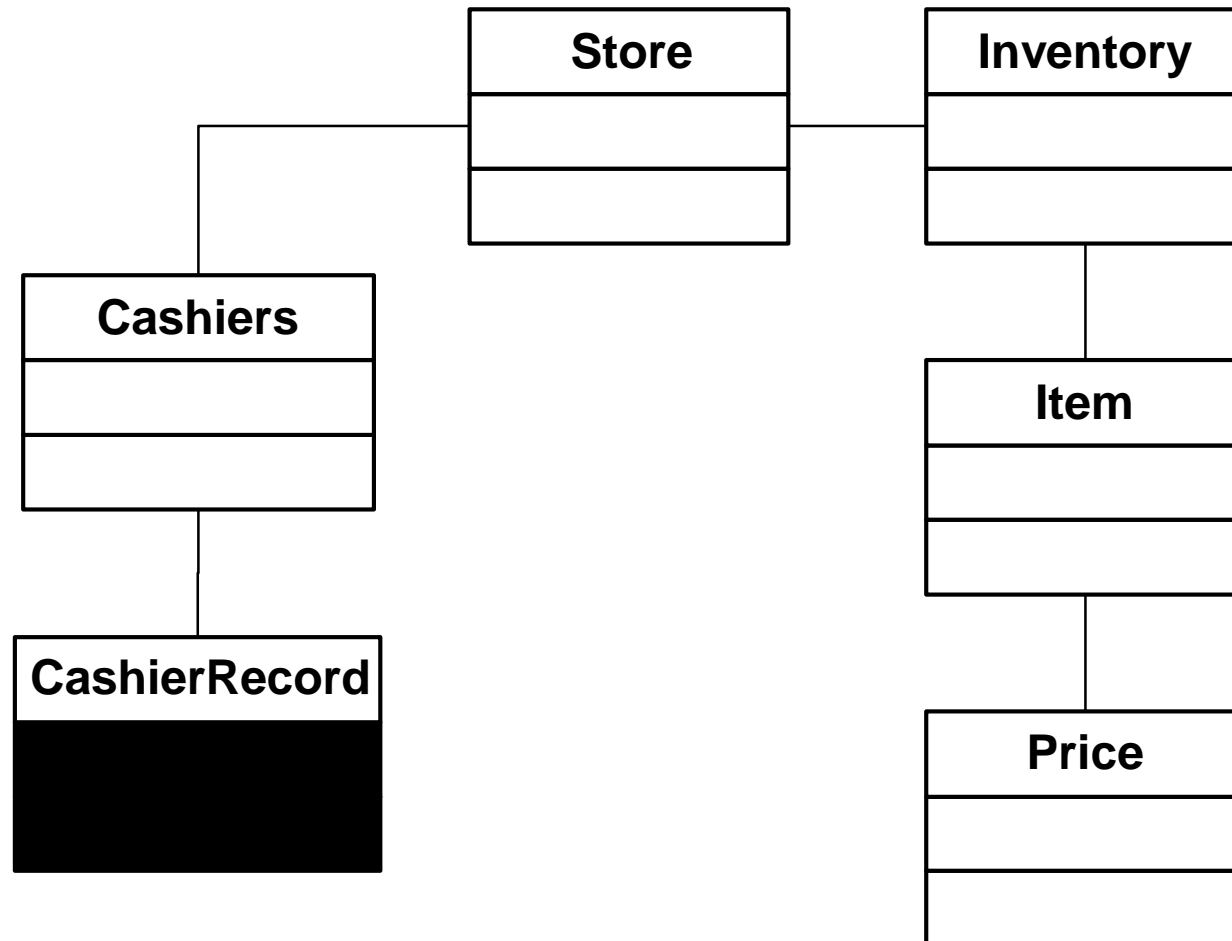
Example: Point of Sale



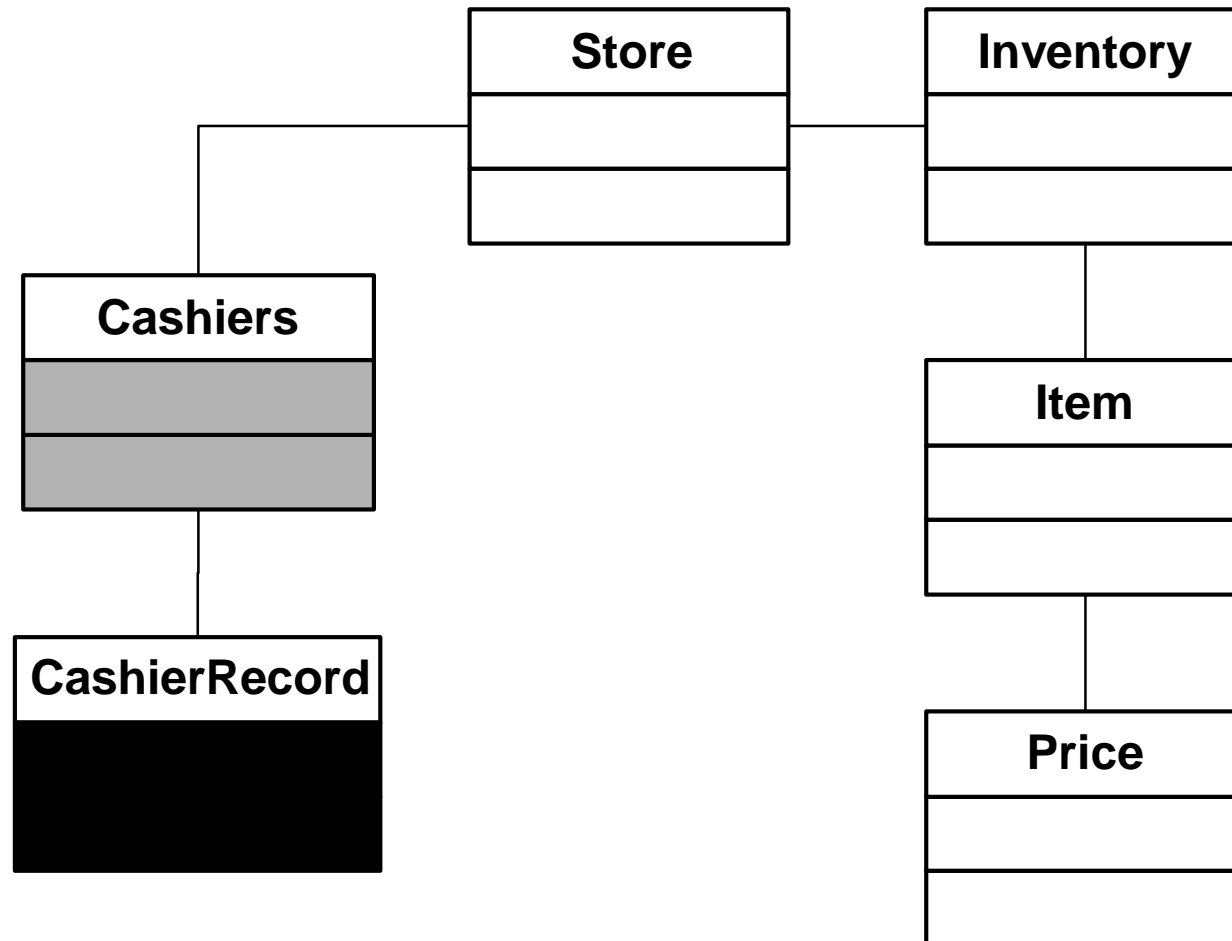
Change request

- ▶ Record cashier sessions
- ▶ A cashier session
 - ▶ total cash and all sales
 - ▶ during the time between the cashier logging in and out.

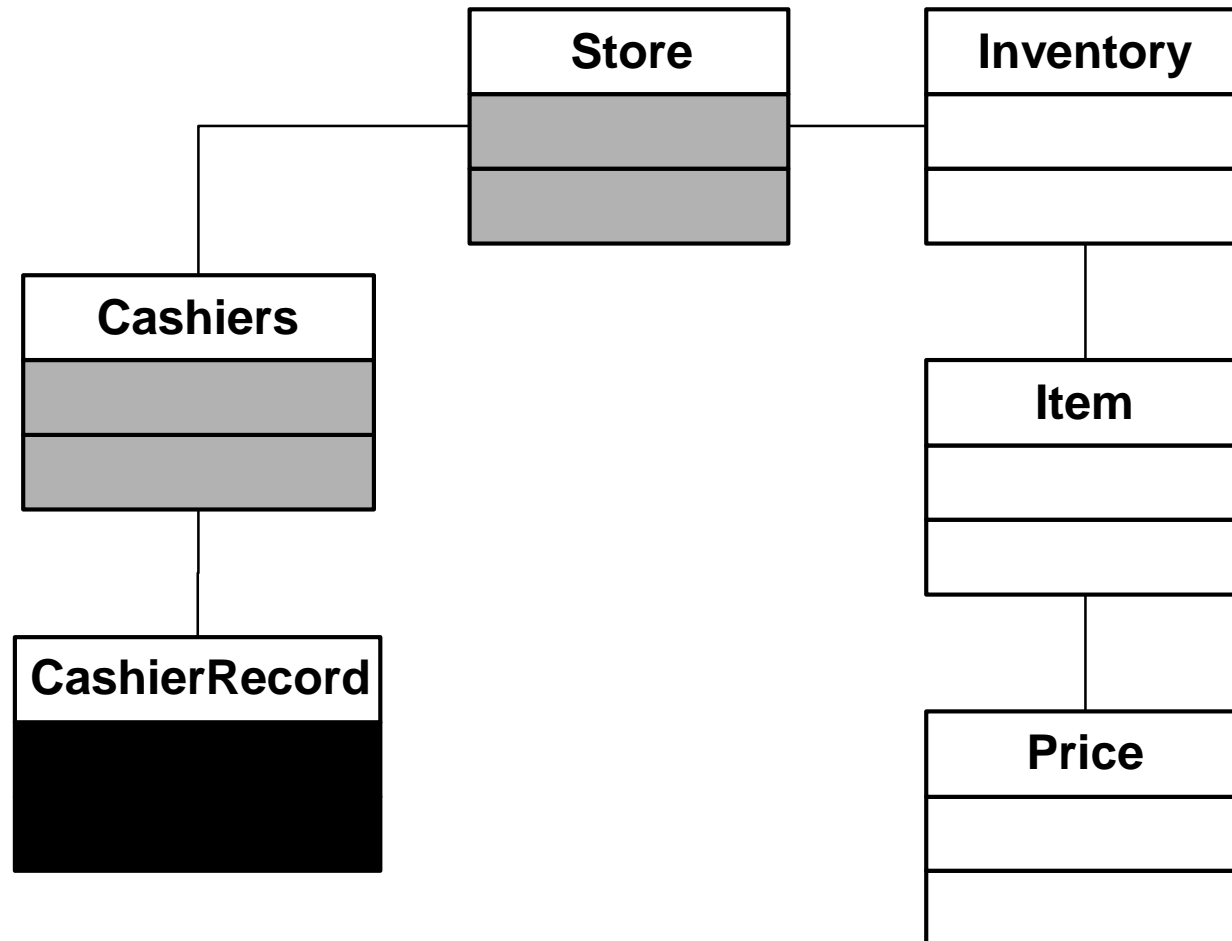
Example: Point of Sale



Example: Point of Sale



Example: Point of Sale



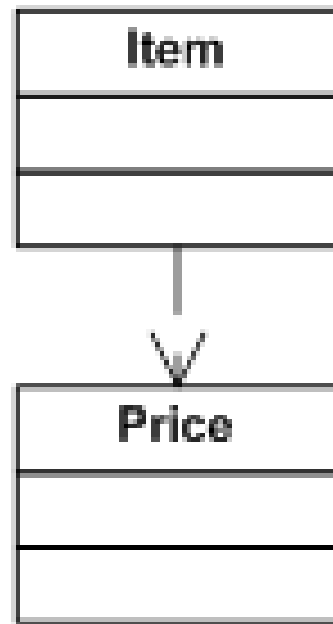
Class interactions

- ▶ **Two classes interact if they have something in common**
 - ▶ One depends on the other
 - ▶ There is a contract between them
 - ▶ They coordinate
 - ▶ They share the same coding, schedule, etc.
- ▶ **Interactions propagate change**
 - ▶ In both directions
 - ▶ From A to B or from B to A

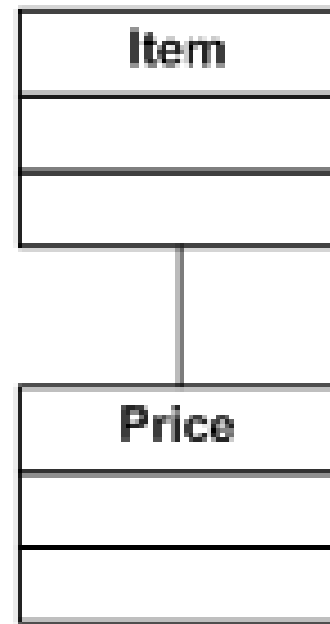
Class Interaction Graph

- ▶ $G = (X, I)$
 - ▶ G ... set of classes
 - ▶ I ... set of interactions
- ▶ **Neighborhood of class A**
 - ▶ $N(A) = \{B \mid (A, B) \in I\}$

Interactions caused by dependencies



Dependency

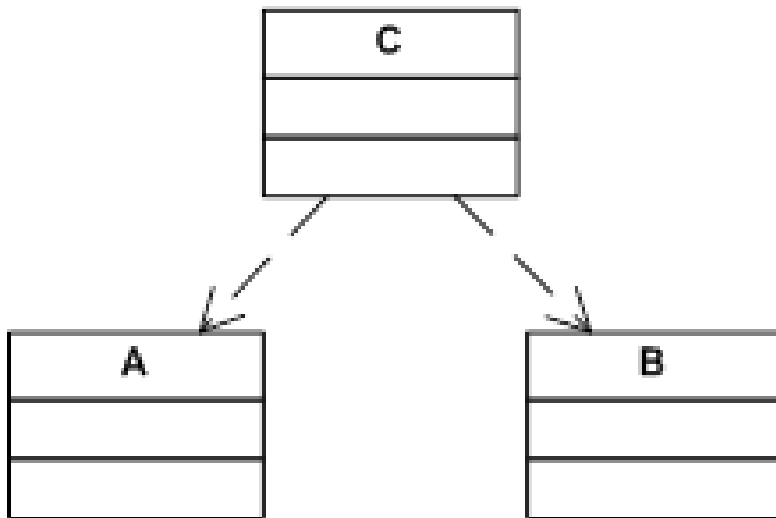


Interaction

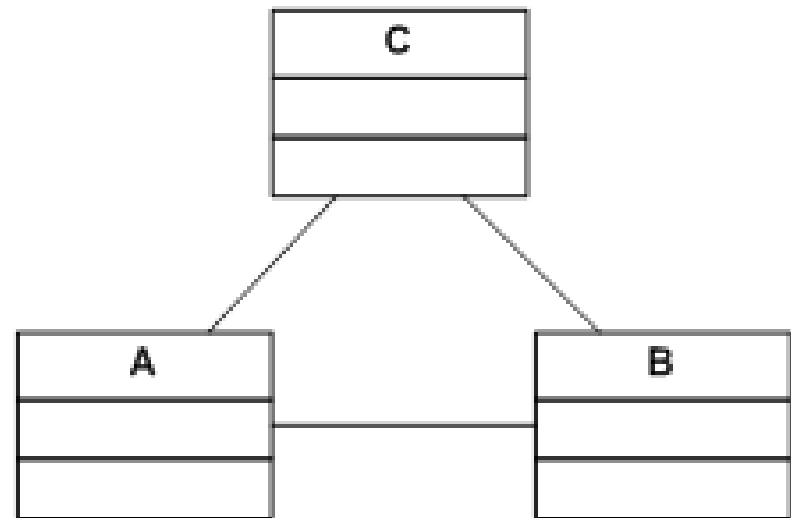
Coordination

```
class C
{
    A  a;        //gets the color code
    B  b;        //paints the screen
    void foo()
    {
        b.paint(a.get());
    }
};
```

Dependency diagram, Interaction diagram

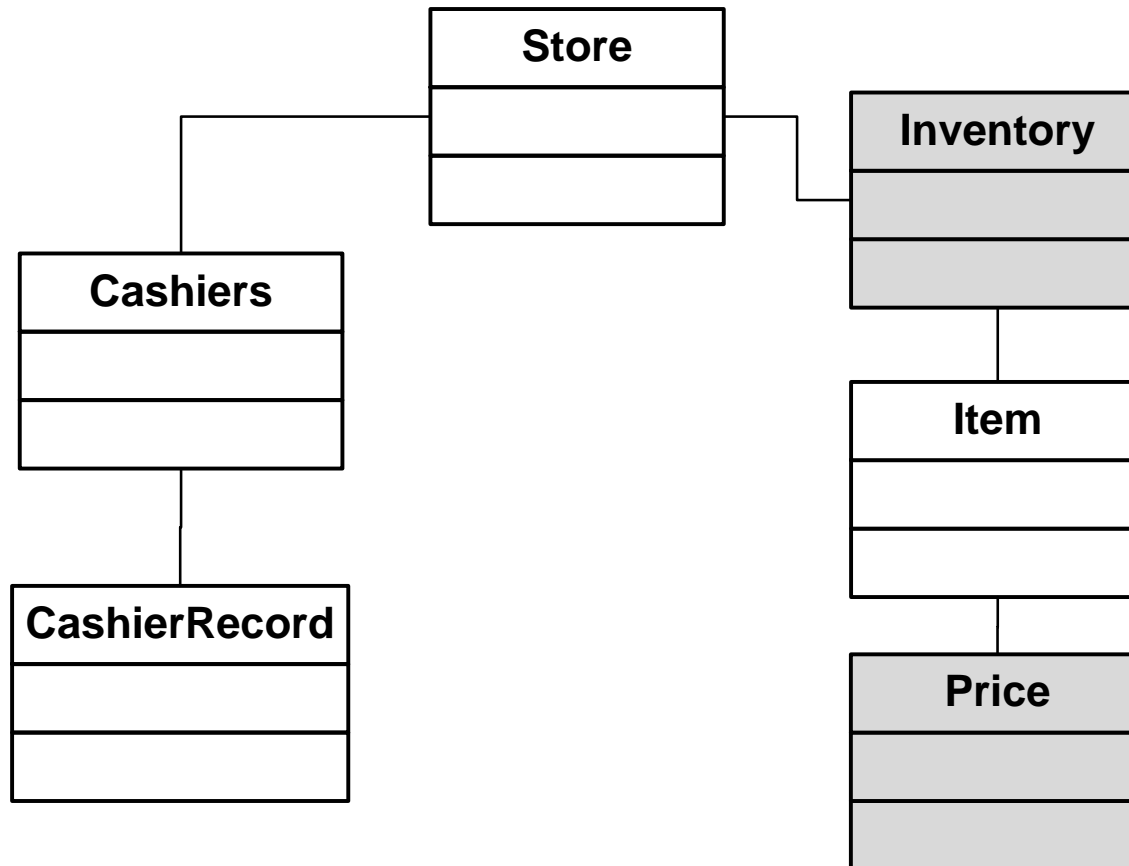


Dependencies



Interactions

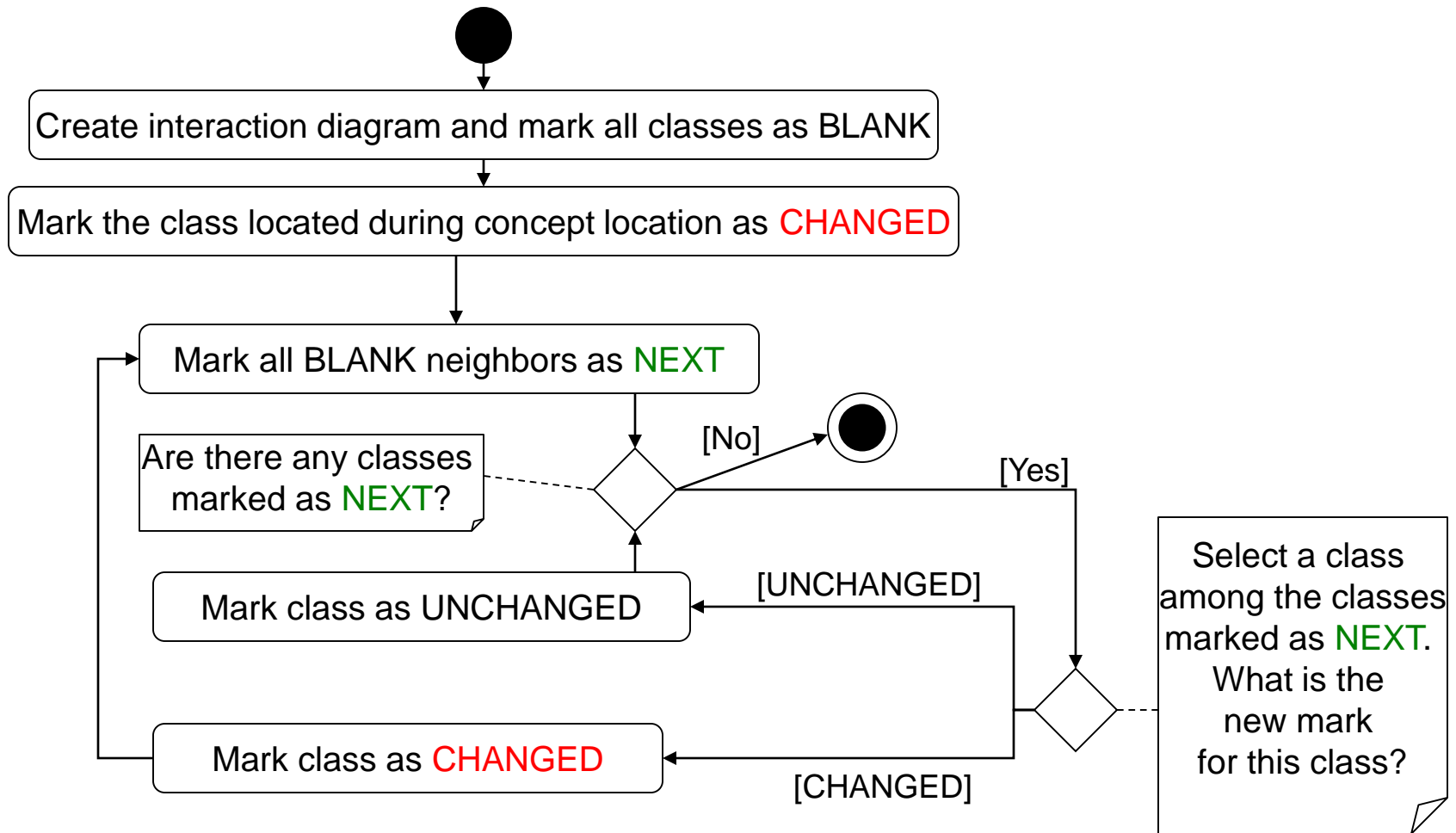
Neighborhood of Item



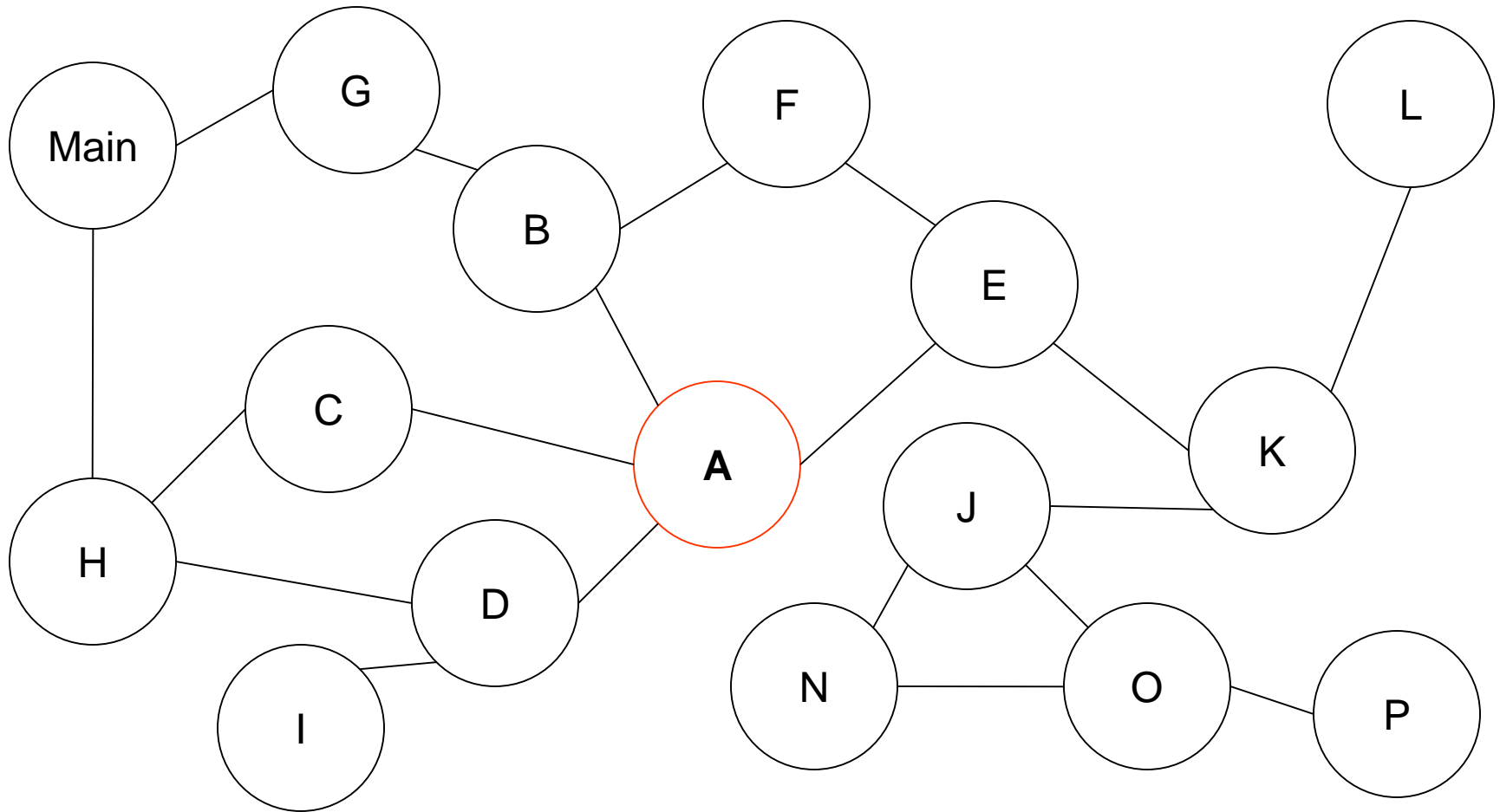
Status of components (marks)

Blank	The class was never inspected and is not scheduled for an inspection.
Changed	The programmers inspected the class and found that it is impacted by the change
Unchanged	The programmers inspected the class and found that it is not impacted by the change.
Next	The class is scheduled for inspection

A simplified IA process



Interactive IA



Color codes

Unknown

Changed

Next

Unchanged

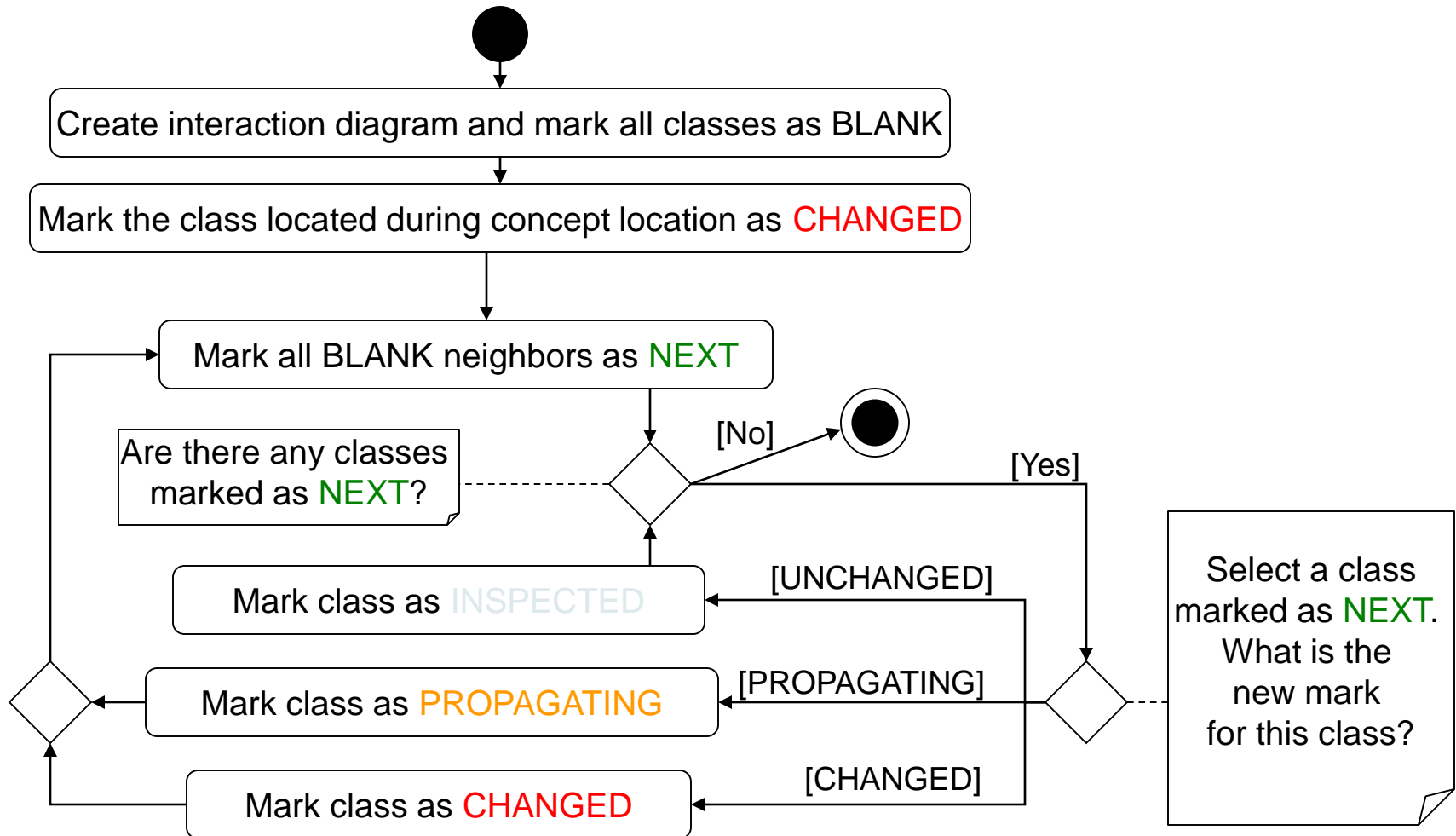
Propagating class: Mailman

- ▶ **John loaned money to Paul**
 - ▶ needs the money back
 - ▶ His situation changed
- ▶ **John writes a letter to Paul**
 - ▶ Mailman takes the letter from John to Paul
 - ▶ Paul must take a part-time job
 - ▶ a big change that propagated from John to Paul

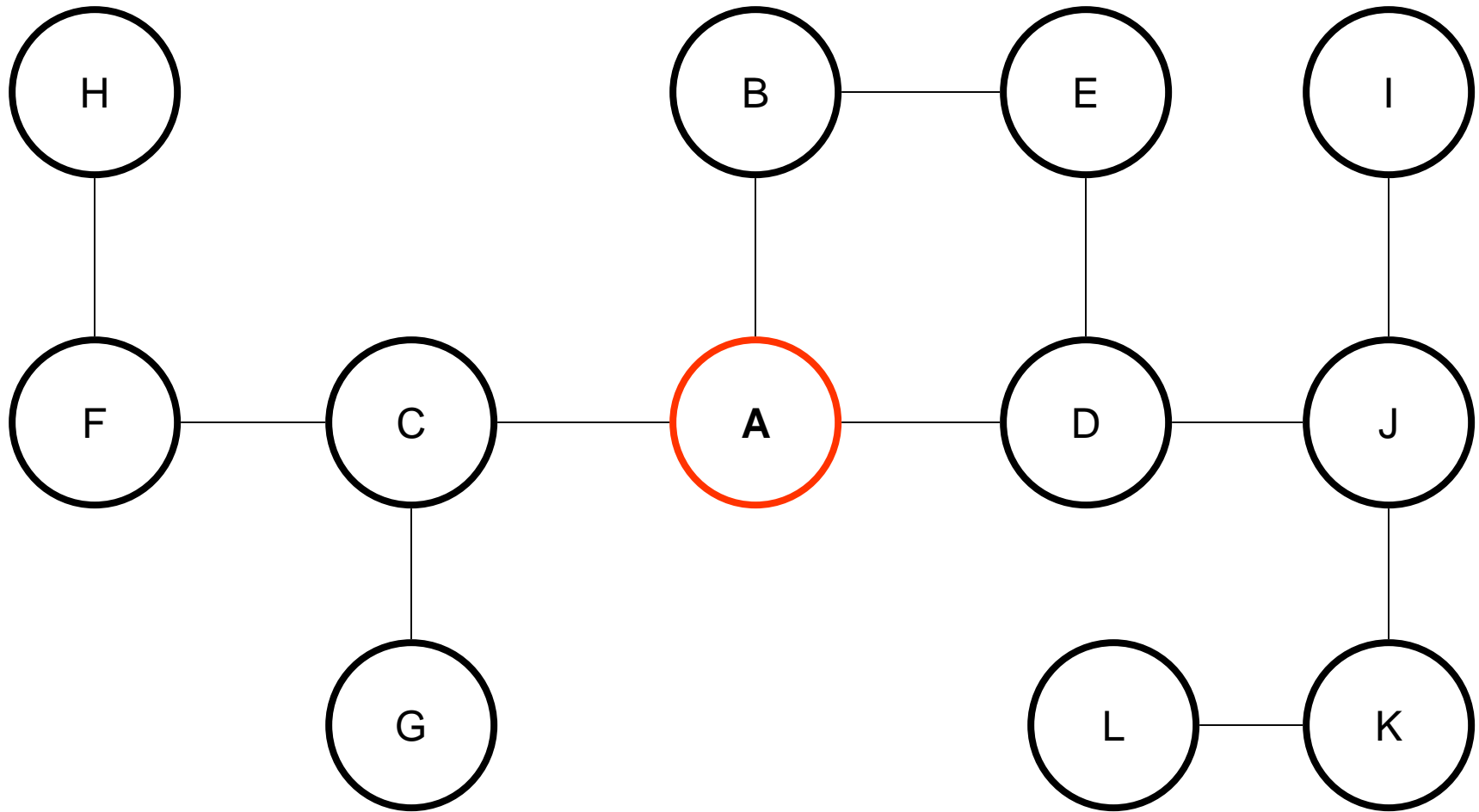
Interactions

- ▶ John interacts with the mailman
- ▶ the mailman interact with Paul
- ▶ The change originated with John and propagates through the mailman to Paul.
 - ▶ mailman is in the middle of the propagation
 - ▶ he does not have to change anything
 - ▶ he just keeps delivering the letters from one person to another.

IA including propagating classes



Iterative IA



Color codes

Unknown

Changed

To be inspected

Inspected and unchanged

Propagating

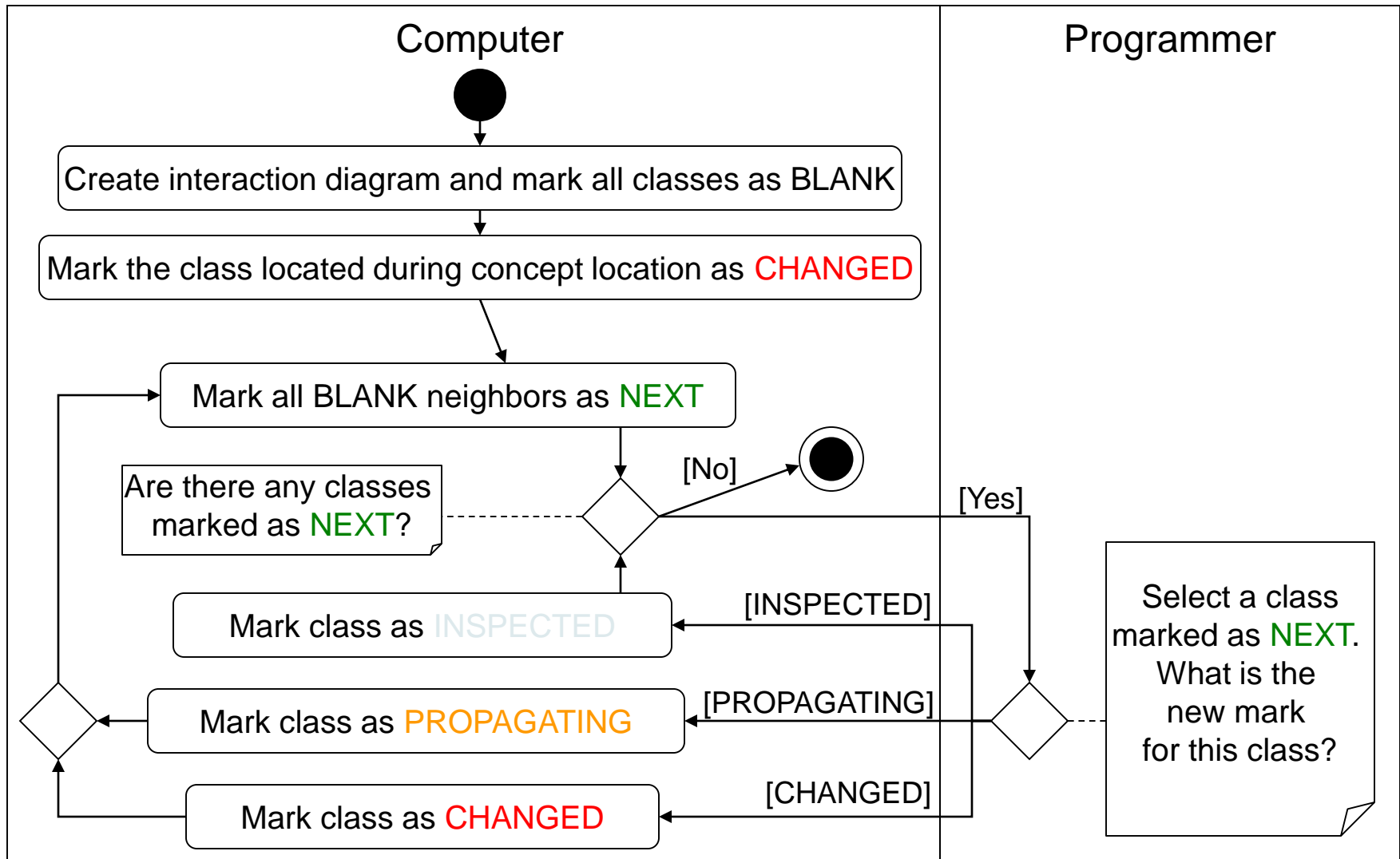
Alternatives in software change

- ▶ Program displaying a temperature in Fahrenheit
 - ▶ change request: display it in Celsius
- ▶ Two separate locations deal with temperature
 - ▶ sensor data converted to the temperature
 - ▶ temperature displayed to the user
- ▶ The change can be done in either place
 - ▶ impact analysis weights these alternatives

The criteria

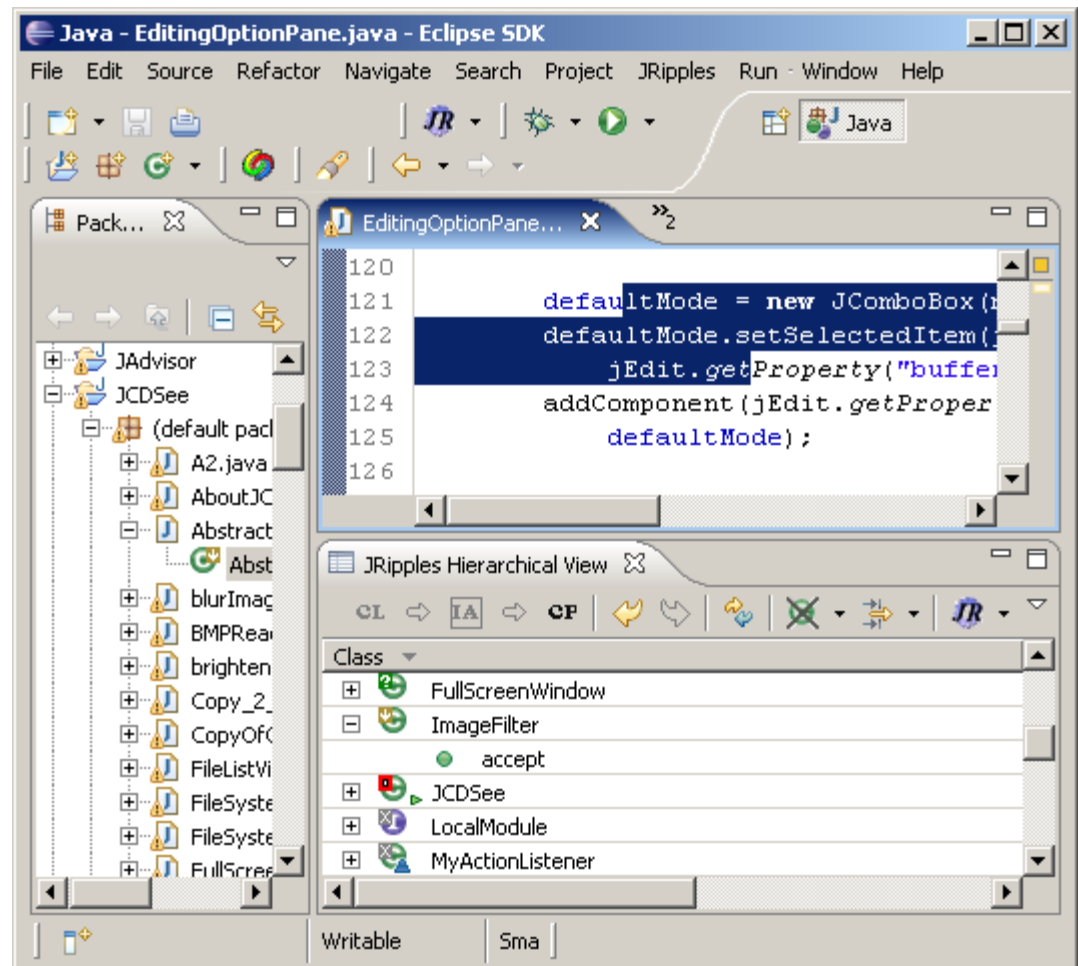
- ▶ Required effort of the change
- ▶ Clarity of the resulting code
- ▶ Often, these two criteria contradict each other
 - ▶ it is easier to adjust the user interface
 - ▶ it is better to have all calculations of the temperature in one place
- ▶ Conflict between short-term and long-term goals

Interactive IA



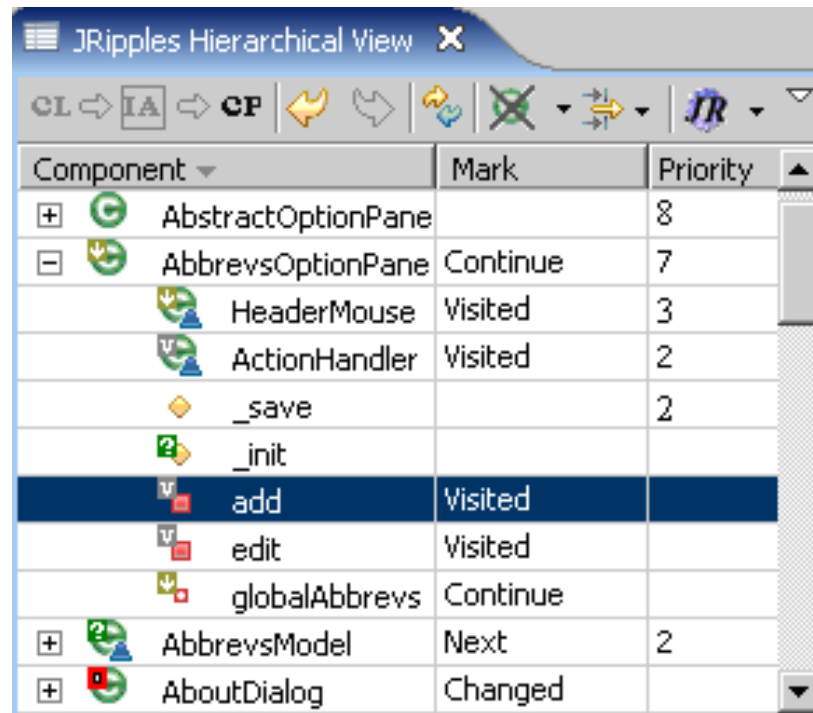
Tool support: JRipples

- ▶ Automatically identifies and suggests components to be inspected **Next**
- ▶ Keeps track of
 - ▶ What was done (**Changed**, **Propagating**, **Unchanged**)
 - ▶ What needs to be done (**Next**)
- ▶ Eclipse plug-in
 - ▶ Java
 - ▶ 15,000 LOC
 - ▶ 150 Classes
 - ▶ 2,500 Methods














<http://jripples.sourceforge.net>

JRipples GUI



The screenshot shows the 'JRipples Hierarchical View' window. It features a toolbar with icons for navigation and editing. Below the toolbar is a table with three columns: 'Component', 'Mark', and 'Priority'. The table lists several components, including 'AbstractOptionPane', 'AbbrevsOptionPane', 'HeaderMouse', 'ActionHandler', '_save', '_init', 'add', 'edit', 'globalAbbrevs', 'AbbrevsModel', and 'AboutDialog'. Each row includes a small icon to the left of the component name, a 'Mark' value, and a 'Priority' value. The 'add' component is highlighted with a dark blue background.

Component	Mark	Priority
 AbstractOptionPane		8
 AbbrevsOptionPane	Continue	7
 HeaderMouse	Visited	3
 ActionHandler	Visited	2
 _save		2
 _init		
 add	Visited	
 edit	Visited	
 globalAbbrevs	Continue	
 AbbrevsModel	Next	2
 AboutDialog	Changed	

JRipples marks and Eclipse

