Software Engineering Project - 1
Schematic Design Of Transistor Level NAND & NOR Gate

Team 26

Members:
Udyan Khurana
Rishabh Sharma
Namita Shelke

Introduction

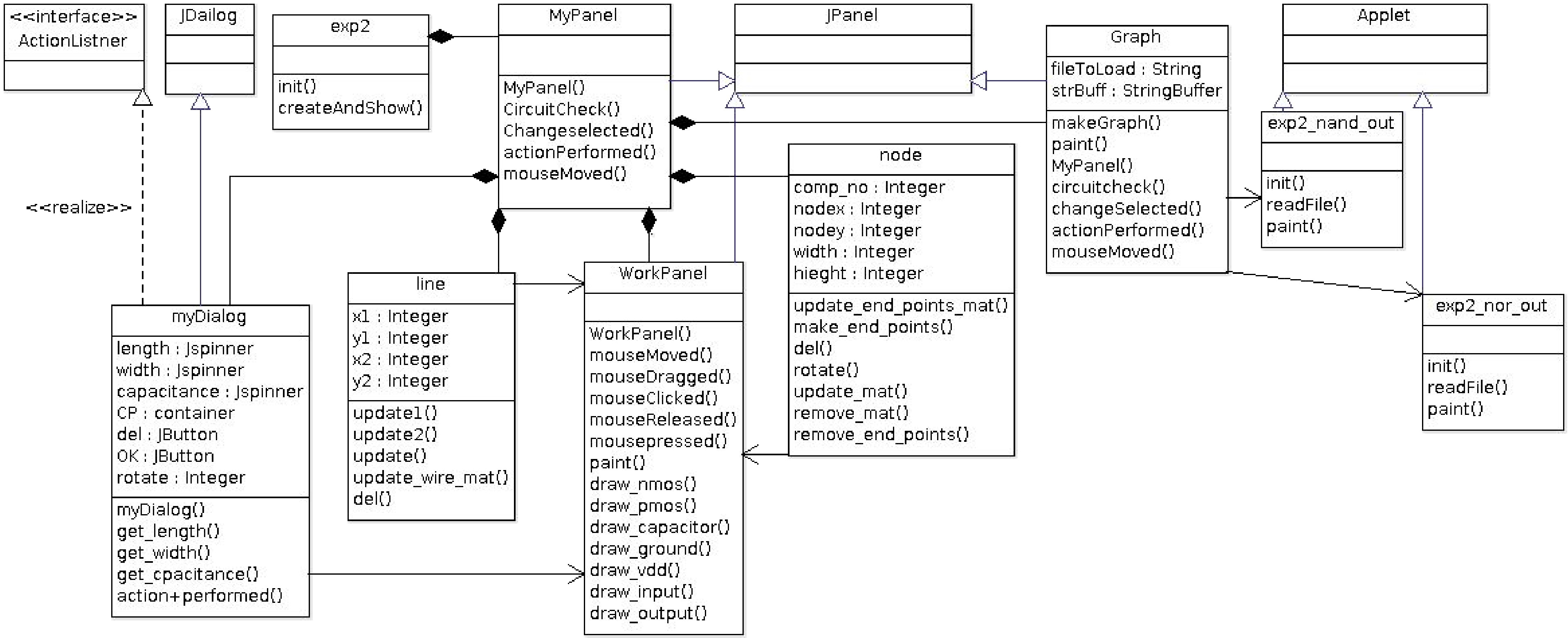
The objective of the experiment is:

- To design a 2 input NAND gate using 2 NMOS and 2 PMOS
- To design a 2 input NOR gate using 2 NMOS and 2 PMOS

Our project focuses on refactoring the current Java based design of the NAND & NOR gate-based circuit maker, and convert it to JavaScript (JS). The conversion to JS is needed as:

- Current experiments are not supported on various browsers.
- Experiments require a plug-in to be installed, impacting UX.
- Code difficult to enhance and maintain.

Current code structure - UML Diagram



Code Smells

Conditional Complexity

- Very large conditional blocks of if-then-else statements.
- Results in increase in LOC (lines of code).
- Such conditional blocks may increase in size later on, making life harder for the person maintaining it!
- SOLUTION: State Pattern - A single class holds the state information and derived classes implement the behaviour shown in each state.

Combinatorial Explosion

- Lots of code doing almost same thing, with just some tiny variations in data.
- SOLUTION: Try to form conditions which can handle all cases in them in 1 logic.

Code Smells (contd.)

Duplicate Code

- Two code fragments that are almost or fully identical.
- In this project, the two files exp2_nand_out.java and exp2_nor_out.java are completely the same except for the class name and the variable “fileToRead”.
- SOLUTION: Make a common file with all the code inside a function and pass it the “fileToRead”.

Uncommented Code

- Too much unexplained code.
- SOLUTION: There should be comments before each class and method explaining what it does.

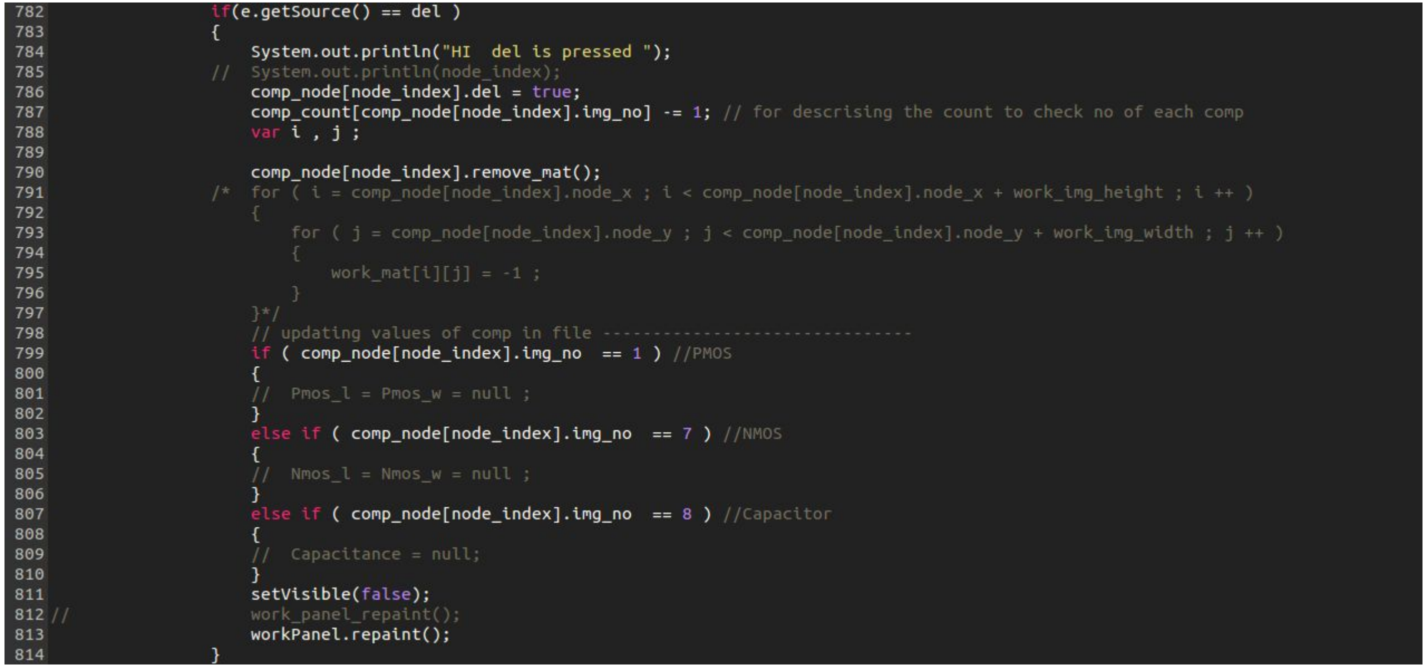
Code Smells (contd.)

Dead Code

- Code that is doing nothing - It does not impact the program in any way.
- For example, if-else conditions with empty blocks. Also, too much code is commented out without any explanation. (See example on Next Page)
- SOLUTIONS:
    - Delete the commented-out code, or give an explanation for why it is still there.
    - Remove the empty if-else blocks if they are doing nothing.

Examples - Dead Code

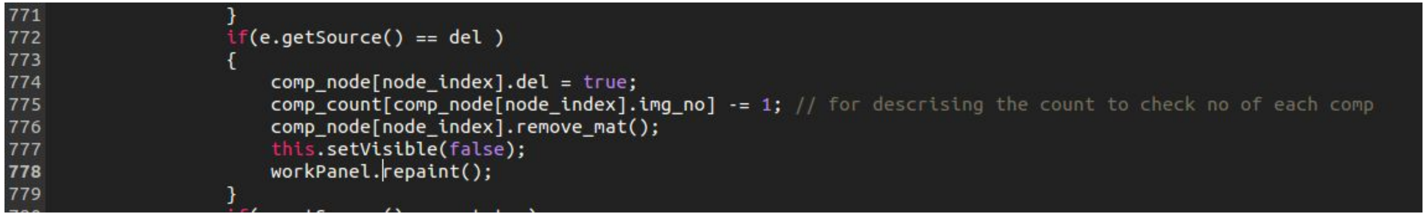
Problem



```
        if(e.getSource() == del )
        {
            System.out.println("HI  del is pressed ");
        //  System.out.println(node_index);
            comp_node[node_index].del = true;
            comp_count[comp_node[node_index].img_no] -= 1; // for descrising the count to check no of each comp
            var i , j ;

            comp_node[node_index].remove_mat();
        /*  for ( i = comp_node[node_index].node_x ; i < comp_node[node_index].node_x + work_img_height ; i ++ )
            {
                for ( j = comp_node[node_index].node_y ; j < comp_node[node_index].node_y + work_img_width ; j ++ )
                {
                    work_mat[i][j] = -1 ;
                }
            }*/
            // updating values of comp in file -------------------------------
            if ( comp_node[node_index].img_no  == 1 ) //PMOS
            {
            //  Pmos_l = Pmos_w = null ;
            }
            else if ( comp_node[node_index].img_no  == 7 ) //NMOS
            {
            //  Nmos_l = Nmos_w = null ;
            }
            else if ( comp_node[node_index].img_no  == 8 ) //Capacitor
            {
            //  Capacitance = null;
            }
            setVisible(false);
//          work_panel_repaint();
            workPanel.repaint();
        }
```

Solution



```
        }
        if(e.getSource() == del )
        {
            comp_node[node_index].del = true;
            comp_count[comp_node[node_index].img_no] -= 1; // for descrising the count to check no of each comp
            comp_node[node_index].remove_mat();
            this.setVisible(false);
            workPanel.repaint();
        }
```

Examples - Combinatorial Explosion

In the class myDialog's constructor, based on the value of the variable "comp_node [node_no].img_no", the behaviour changes slightly. We can instead try to make a common method for specifying a group of constraints with variable values and call that again.

Also, in the function actionPerformed(), the value of the variable "comp_node [node_index].comp_no" tells if the value of Pmos1 variables or Pmos2 variables will change. Instead, we can change the value of Pmos[comp_node[node_index]. comp_no], where Pmos is an array of strings.

Thank You