First let's talk about the reused objects:

- TDistribution object
  - We will use it exactly the same
- Simpson object
  - We will use it exactly the same
- some parts of the Main object (GUI table)
  - For create the CalculationP object
- GUI object
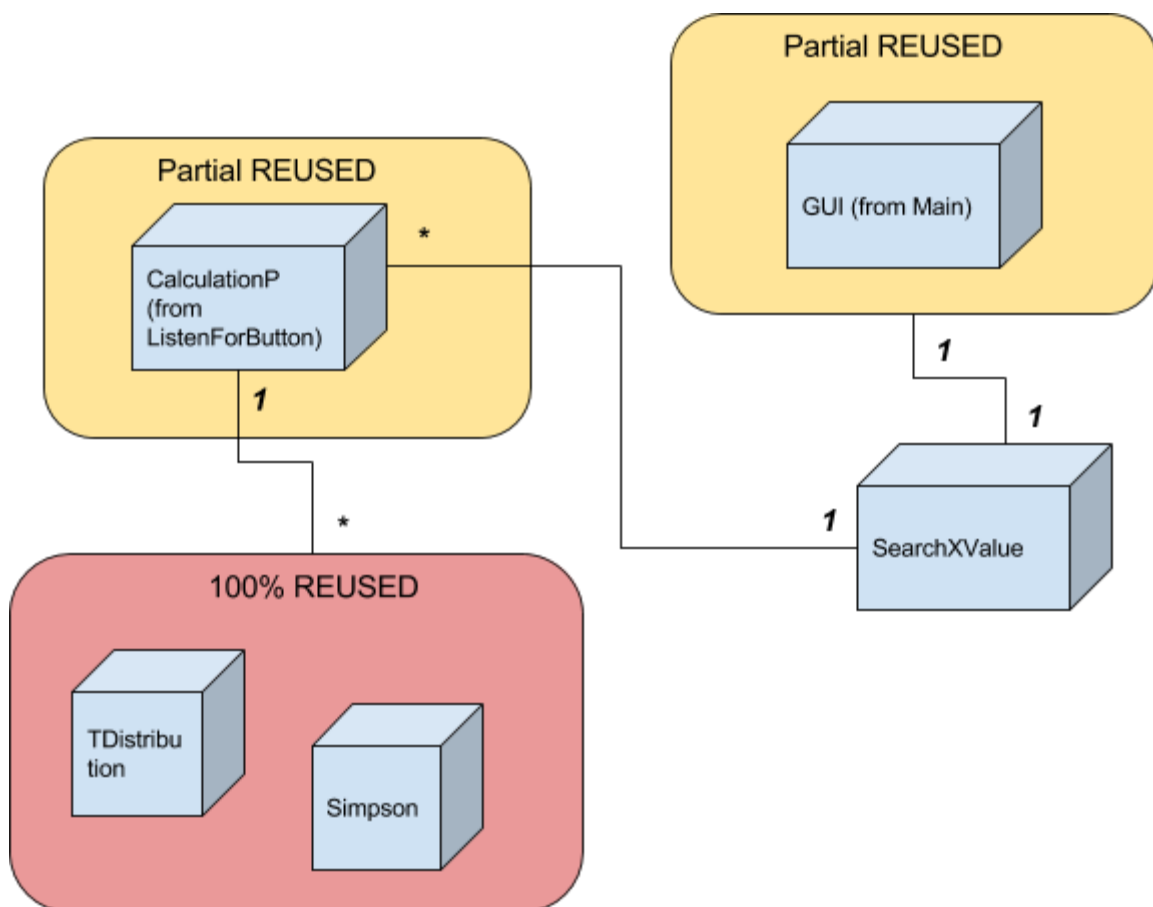 - Will be a subtract from the Main class of the previous program.

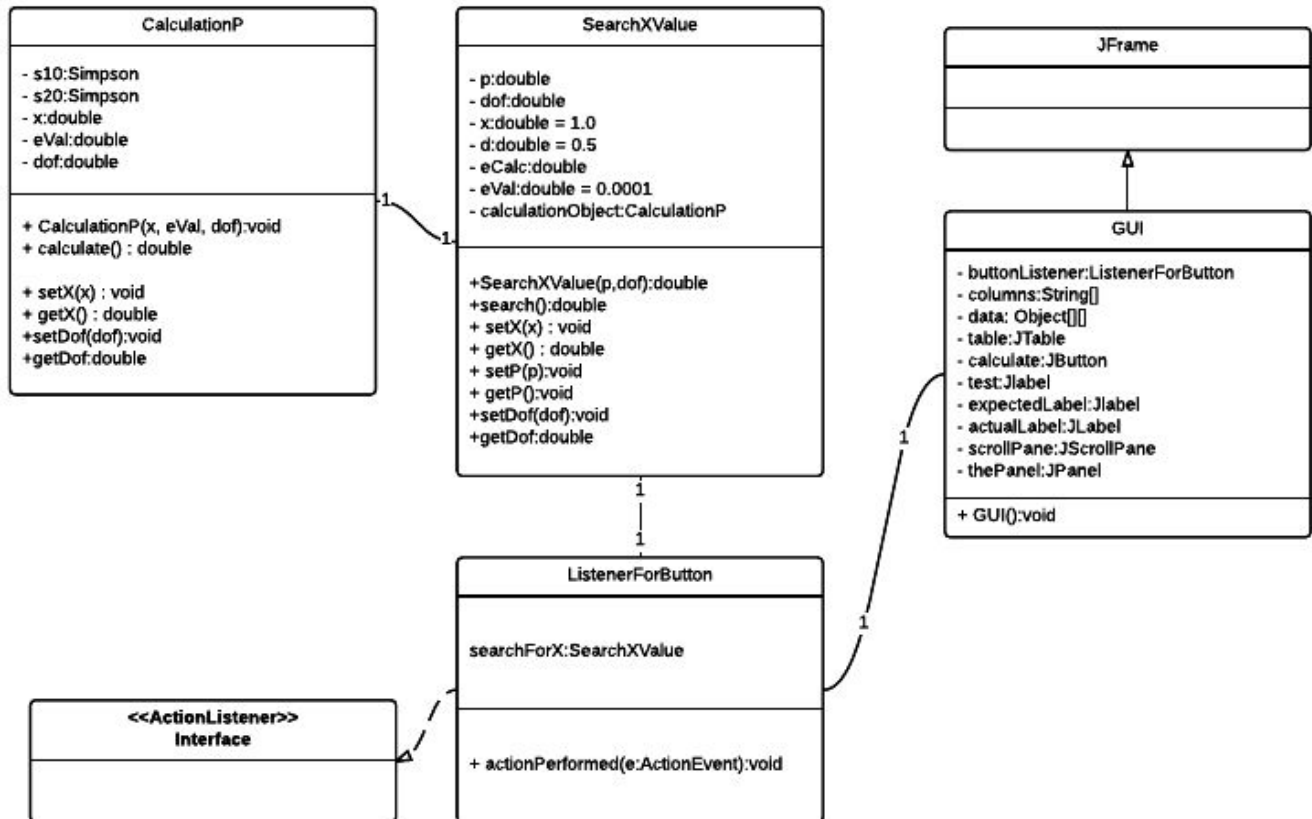We will modify the GUI to meet the new requirements.

New creations:
- SearchXValue object

The new program will show a GUI with the static 3 test cases, and a button that will display the expected value for X.
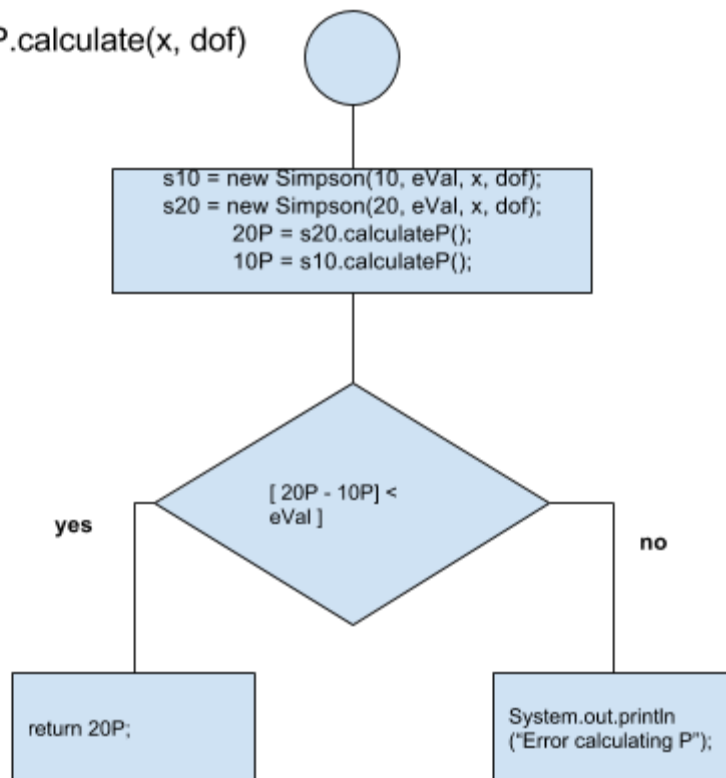
The diagram of the interaction of objects will be:

A more detailed class diagram can be found here:

CalculationP

- s10:Simpson
- s20:Simpson
- x:double
- eVal:double
- dof:double

+ CalculationP(x, eVal, dof):void
+ calculate() : double

+ setX(x) : void
+ getX() : double
+setDof(dof):void
+getDof:double

SearchXValue

- p:double
- dof:double
- x:double = 1.0
- d:double = 0.5
- eCalc:double
- eVal:double = 0.0001
- calculationObject:CalculationP

+SearchXValue(p,dof):double
+search():double
+ setX(x) : void
+ getX() : double
+ setP(p):void
+ getP():void
+setDof(dof):void
+getDof:double

JFrame

GUI

- buttonListener:ListenerForButton
- columns:String[]
- data: Object[][]
- table:JTable
- calculate:JButton
- test:Jlabel
- expectedLabel:Jlabel
- actualLabel:JLabel
- scrollPane:JScrollPane
- thePanel:JPanel

+ GUI():void

ListenerForButton

searchForX:SearchXValue

+ actionPerformed(e:ActionEvent):void

<<ActionListener>>
Interface

CalculationP.calculate(x, dof)

s10 = new Simpson(10, eVal, x, dof);
s20 = new Simpson(20, eVal, x, dof);
20P = s20.calculateP();
10P = s10.calculateP();

[20P - 10P] < eVal]

yes

no

return 20P;

System.out.println
("Error calculating P");

SearchXValue.search(): double

p
dof
x = 1
d = 0.5
eVal = 0.0001
calculationObject(x,eVal,dof)
theValue = calculationObject.calculate()

No — theValue == this.p — **Yes**

return X value

Yes — theValue < this.p — **No**

x = x + d

x = x - d

calculationObject().setX(this.x)
theValue = calculationObject.calculate()
eCalc = this.p - theValue

No — eCalc == eVal — **Yes**

Yes — Math.signum(eCal) == Math.signum(eVal) — **No**

d = d/2

d = 2

Yes — d <= 0 — **No**