

A taxonomy of fix patterns (FPs)

Nowack et al.

The taxonomy contains **TBar** fix patterns and the fix patterns generated using extended **Sun** and **Java** parsers.

FP1. Insert Cast Checker.

```
+ if (exp instanceof T) {  
    var = (T) exp; .....  
+ }
```

FP2. Insert Null Pointer Checker.

```
FP2.1: + if (exp != null) {  
    ...exp...;  
    + }  
FP2.2: + if (exp == null) return DEFAULT_VALUE;  
    ...exp...;  
FP2.3: + if (exp == null) exp = exp1;  
    ...exp...;  
FP2.4: + if (exp == null) continue;  
    ...exp...;  
FP2.5: + if (exp == null)  
    +     throw new IllegalArgumentException(...);  
    ...exp...;  
FP2.6: + if (exp == null) return literal;  
    FP2.6.1: + if (exp == null) return null;
```

FP3. Insert Range Checker.

```
+ if (index < exp.length) {  
    ...exp[index]...; .....  
+ }  
OR  
+ if (index < exp.size()) {  
    ...exp.get(index)...; .....  
+ }
```

FP4. Insert Missed Statement.

```
FP4.1: + method(exp);  
FP4.2: + return DEFAULT_VALUE;  
FP4.3: + try {  
    statement; .....  
    + } catch (Exception e) { ... }  
FP4.4: + if (conditional_exp) {  
    statement; .....  
    + }  
    FP4.4.1: + if (var != 0) {  
        statement; ...  
        + }  
    FP4.4.2: + if (conditional_exp) {  
        + statement; ...  
        + }  
    FP4.4.3: + if (conditional_exp) {  
        + } else {  
        + statement; ...  
        + }  
FP4.5: + var1 = var2  
    FP4.5.1: + var = literal
```

FP4.6: + obj.var = var
FP4.7: + T var = literal
FP4.8: + T var
FP4.9: + **case** exp
FP4.10: - var
+ var = literal

FP5. Mutate Class Instance Creation.

```
public Object clone() {
-   ... new T();
+   ... (T) super.clone();
}
```

FP6. Mutate Conditional Expression.

FP6.1: - ...condExp1...
+ ...condExp2...
FP6.2: - ...condExp1 Op condExp2...
+ ...condExp1...
FP6.3: - ...condExp1...
+ ...condExp1 Op condExp2...

FP7. Mutate Data Type.

FP7.1: - T1 var ...;
+ T2 var ...;
FP7.2: - ...(T1) exp...;
+ ...(T2) exp...;

FP8. Mutate Integer Division Operation.

FP8.1: - ...dividend / divisor...
+ ...dividend / (**double** or **float**) divisor...
FP8.2: - ...dividend / divisor...
+ ...(**double** or **float**) dividend / divisor...
FP8.3: - ...dividend / divisor...
+ ...(1.0 / divisor) * dividend...

FP9. Mutate Literal Expression.

FP9.1: - ...literal1...
+ ...literal2...
FP9.2: - ...literal1...
+ ...exp...

FP10. Mutate Method Invocation Expression.

FP10.1: - ...method1(args)...
+ ...method2(args)...
FP10.1.1: - method1()
+ method2().method3()
FP10.2: - ...method1(arg1, arg2, ...)...
+ ...method1(arg1, arg3, ...)...
FP10.3: - ...method1(arg1, arg2, ...)...
+ ...method1(arg1, ...)...
FP10.4: - ...method1(arg1, ...)...
+ ...method1(arg1, arg2, ...)...
FP10.4.1: - method1()
+ method1(arg)
FP10.5: - var
+ method
FP10.6: - method1().method2()
+ method3()

FP11. Mutate Operator.

FP11.1: - ...exp1 Op1 exp2...
+ ...exp1 Op2 exp2...
FP11.2: - ...(exp1 Op1 exp2) Op2 exp3...
+ ...exp1 Op1 (exp2 Op2 exp3)...

FP11.3: - ...exp instanceof T...

+ ...exp != null...

FP11.4: - ... exp

+ ... method

FP12. Mutate Return Statement.

- return exp1;

+ return exp2;

FP13. Mutate Variable.

FP13.1: - ...var1...

+ ...var2...

FP13.1.1: - obj.attr

+ var

FP13.1.2: - obj1.method()

+ obj2.method()

FP13.1.3: - obj1.method1(args)

+ obj2.method2().method1(args)

FP13.1.4: - obj1.method1()

+ obj2.method2(obj1)

FP13.1.5: - obj1.method1(obj2)

+ obj3.method2().method3(obj1, obj2)

FP13.2: - ...var1...

+ ...exp...

FP13.3: - ...var...

+ ...new instance (wrapping var)...

FP13.4: - ...var...

+ ... method(var, ...)...

FP14. Move Statement.

- statement;

.....

+ statement;

FP15. Remove Buggy Statement.

FP15.1: -

- statement;

.....

FP15.1.1: - method()

FP15.2: - methodDeclaration(Arguments) {

-; statement;.....

- }

FP16. Mutate Method Definition.

- modifier T method(T1 arg1, ...)...

+ modifier T method(T2 arg2, ...)...