**Algorithm design**

The algorithm is based on control flow graph (CFG). CFG is a graph G<V, E>. V represents the set of nodes, which contains “Basic Blocks” and “Decision Nodes”. E represents the jumping relation between nodes.

Fault localization will first determine which basic block is most suspicious. Then the fix algorithm will generate fix suggestions based on the CONTEXT of the basic block. By CONTEXT, we mean the syntactical context of the basic block. For example:

If(expression){

Statement1;

} else {

Statement2;

}

Statement1 is the “Statement block” of an IfStatement. Then if we try to fix Statement1, we put it into the context of an IfStatement and apply fix strategies for IfStatements.

Our fix algorithm requires strategies for all possible contexts of basic blocks. Therefore we need to enumerate all legal syntactic structures that may affect the correctness of the program, and design reasonable fix patterns for each of them. JDT models syntactic structures as ASTNodes, so this is where we start.

This is the ASTNode type tree. Red means we need to design fix algorithms for them. Black means it has nothing to do with our fix. Green means they are relative to the execution process of program, but we are currently unable to deal with them.

* + [ASTNode](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ASTNode.html)
    - [AnonymousClassDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AnonymousClassDeclaration.html)
    - [BodyDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/BodyDeclaration.html)
      * [AbstractTypeDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AbstractTypeDeclaration.html)
        + [AnnotationTypeDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AnnotationTypeDeclaration.html)
        + [EnumDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EnumDeclaration.html)
        + [TypeDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeDeclaration.html)
      * [AnnotationTypeMemberDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AnnotationTypeMemberDeclaration.html)
      * [EnumConstantDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EnumConstantDeclaration.html)
      * [FieldDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/FieldDeclaration.html)
      * [Initializer](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Initializer.html)
      * [MethodDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MethodDeclaration.html)
    - [CatchClause](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/CatchClause.html)
    - [Comment](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Comment.html)
      * [BlockComment](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/BlockComment.html)
      * [Javadoc](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Javadoc.html)
      * [LineComment](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/LineComment.html)
    - [CompilationUnit](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/CompilationUnit.html)
    - [Dimension](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Dimension.html)
    - [Expression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Expression.html)
      * [Annotation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Annotation.html) (implements [IExtendedModifier](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IExtendedModifier.html))
        + [MarkerAnnotation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MarkerAnnotation.html)
        + [NormalAnnotation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/NormalAnnotation.html)
        + [SingleMemberAnnotation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SingleMemberAnnotation.html)
      * [ArrayAccess](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ArrayAccess.html)
      * [ArrayCreation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ArrayCreation.html)
      * [ArrayInitializer](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ArrayInitializer.html)
      * [Assignment](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Assignment.html)
      * [BooleanLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/BooleanLiteral.html)
      * [CastExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/CastExpression.html)
      * [CharacterLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/CharacterLiteral.html)
      * [ClassInstanceCreation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ClassInstanceCreation.html)
      * [ConditionalExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ConditionalExpression.html)
      * [FieldAccess](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/FieldAccess.html)
      * [InfixExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/InfixExpression.html)
      * [InstanceofExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/InstanceofExpression.html)
      * [LambdaExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/LambdaExpression.html)
      * [MethodInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MethodInvocation.html)
      * [MethodReference](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MethodReference.html)
        + [CreationReference](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/CreationReference.html)
        + [ExpressionMethodReference](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ExpressionMethodReference.html)
        + [SuperMethodReference](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SuperMethodReference.html)
        + [TypeMethodReference](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeMethodReference.html)
      * [Name](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Name.html) (implements [IDocElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IDocElement.html))
        + [QualifiedName](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/QualifiedName.html)
        + [SimpleName](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SimpleName.html)
      * [NullLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/NullLiteral.html)
      * [NumberLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/NumberLiteral.html)
      * [ParenthesizedExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ParenthesizedExpression.html)
      * [PostfixExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/PostfixExpression.html)
      * [PrefixExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/PrefixExpression.html)
      * [StringLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/StringLiteral.html)
      * [SuperFieldAccess](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SuperFieldAccess.html)
      * [SuperMethodInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SuperMethodInvocation.html)
      * [ThisExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ThisExpression.html)
      * [TypeLiteral](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeLiteral.html)
      * [VariableDeclarationExpression](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/VariableDeclarationExpression.html)
    - [ImportDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ImportDeclaration.html)
    - [MemberRef](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MemberRef.html) (implements [IDocElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IDocElement.html))
    - [MemberValuePair](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MemberValuePair.html)
    - [MethodRef](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MethodRef.html) (implements [IDocElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IDocElement.html))
    - [MethodRefParameter](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/MethodRefParameter.html)
    - [Modifier](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Modifier.html) (implements [IExtendedModifier](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IExtendedModifier.html))
    - [PackageDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/PackageDeclaration.html)
    - [Statement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Statement.html)
      * [AssertStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AssertStatement.html)
      * [Block](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Block.html)
      * [BreakStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/BreakStatement.html)
      * [ConstructorInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ConstructorInvocation.html)
      * [ContinueStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ContinueStatement.html)
      * [DoStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/DoStatement.html)
      * [EmptyStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EmptyStatement.html)
      * [EnhancedForStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EnhancedForStatement.html)
      * [ExpressionStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ExpressionStatement.html)
      * [ForStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ForStatement.html)
      * [IfStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IfStatement.html)
      * [LabeledStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/LabeledStatement.html)
      * [ReturnStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ReturnStatement.html)
      * [SuperConstructorInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SuperConstructorInvocation.html)
      * [SwitchCase](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SwitchCase.html)
      * [SwitchStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SwitchStatement.html)
      * [SynchronizedStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SynchronizedStatement.html)
      * [ThrowStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ThrowStatement.html)
      * [TryStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TryStatement.html) //???????
      * [TypeDeclarationStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeDeclarationStatement.html)
      * [VariableDeclarationStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/VariableDeclarationStatement.html)
      * [WhileStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/WhileStatement.html)
    - [TagElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TagElement.html) (implements [IDocElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IDocElement.html))
    - [TextElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TextElement.html) (implements [IDocElement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IDocElement.html))
    - [Type](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Type.html)
      * [AnnotatableType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/AnnotatableType.html)
        + [NameQualifiedType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/NameQualifiedType.html)
        + [PrimitiveType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/PrimitiveType.html)
        + [QualifiedType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/QualifiedType.html)
        + [SimpleType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SimpleType.html)
        + [WildcardType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/WildcardType.html)
      * [ArrayType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ArrayType.html)
      * [IntersectionType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IntersectionType.html)
      * [ParameterizedType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ParameterizedType.html)
      * [UnionType](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/UnionType.html)
    - [TypeParameter](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeParameter.html)
    - [VariableDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/VariableDeclaration.html)
      * [SingleVariableDeclaration](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SingleVariableDeclaration.html)
      * [VariableDeclarationFragment](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/VariableDeclarationFragment.html)

According to this type tree, only these following structures can provide a “context” within a method declaration aside from the method itself:

[Block](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/Block.html)

[DoStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/DoStatement.html)

[EnhancedForStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EnhancedForStatement.html)

[ForStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ForStatement.html)

[IfStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/IfStatement.html)

[SwitchCase](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SwitchCase.html)

[SwitchStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SwitchStatement.html)

[TryStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TryStatement.html) //???????

[WhileStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/WhileStatement.html)

We call these structures “compound statements”.

The following statements will be the statements constructing each basic block:

[BreakStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/BreakStatement.html)

[ConstructorInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ConstructorInvocation.html)

[ContinueStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ContinueStatement.html)

[EmptyStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/EmptyStatement.html)

[ExpressionStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ExpressionStatement.html)

[LabeledStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/LabeledStatement.html)

[ReturnStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/ReturnStatement.html)

[SuperConstructorInvocation](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/SuperConstructorInvocation.html)

[TypeDeclarationStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/TypeDeclarationStatement.html)

[VariableDeclarationStatement](http://help.eclipse.org/mars/topic/org.eclipse.jdt.doc.isv/reference/api/org/eclipse/jdt/core/dom/VariableDeclarationStatement.html)

We call these structures “simple statements”.

Notice that Assignment is treated as an expression and included in “ExpressionStatement”.

Blocks within compound statements would first considered to be structurally transformed. Then simple statements within blocks will be transformed according to their unique types.

**Structural transformation**

Structural transformation should consider 1) the syntactic structure context, 2) information coming from fault localization.

Fault localization will identify the basic blocks that needs to be modified. They can be either execution blocks or decision blocks. For execution blocks, they contain only sequential simple statements. There are three general kind of contexts:

1. (Sequence) Directly in a method body.

In this case, it should contain simple statements except ContinueStatement and BreakStatement.

1. (Branch) Wrapped in an IfStatement or a SwitchCase.
2. (Loop) Wrapped in a DoStatement, an EnhancedStatement, a ForStatement or a WhileStatement.

Structural transformation for blocks basically contains 3 atomic operations:

1. Swap: change the order of two statements
2. Delete: delete one statement
3. Insert: insert one statement
   1. Insert a new ifStatement
   2. Insert a new assignment

If it is in the context of Branch or Loop, we have:

1. Move out: move a statement outside the Branch or Loop to its wrapper.
2. Move in: move a statement inside the Branch or Loop from its wrapper.
3. Jump Introduction: insert BreakStatements or ContinueStatements

Now we discuss specific structural transformation strategies for different syntactical context.

1. Block directly under MethodDeclarationBlock
2. Insert
   1. BranchIntroduction
      1. NullChecker
      2. CastChecker
      3. General Branch Introduction: if(?){??}
   2. Assignment

?=??

1. Delete

If two statements look very similar, delete one of them.

1. Swap

If two statements look very similar, swap them.

1. Block in IfStatement
2. Insert
   1. BranchIntroduction
      1. NullChecker
      2. CastChecker
      3. General Branch Introduction: if(?){??}
   2. Assignment

?=??

1. Swap out

Choose a statement and swap it out in parallel branches

* 1. If block is in if(){here}else{}, then swap one statement in {here} with else{}

1. Block in SwitchCase (SwitchStatement is composed of SwitchCases)
2. Insert
   1. BranchIntroduction
      1. NullChecker
      2. CastChecker
      3. General Branch Introduction: if(?){??}
   2. Assignment

?=??

* 1. JumpIntroduction
     1. Insert BreakStatement at the end
     2. Insert ReturnStatement at the end

1. Block in DoStatement
2. Move in
3. Move out
4. Remove
5. Block in EnhancedForStatement
6. Branch Introduction
   1. Exit Condition
   2. BreakStatement
   3. ContinueStatement
   4. General Branch Introduction
7. Move in
8. Block in ForStatement
9. Branch Introduction
   1. BreakStatement
   2. ContinueStatement
   3. General Branch Introduction
10. Move in
11. Block WhileStatement
12. Branch Introduction
    1. BreakStatement / If(Expression){Break;}
    2. ContinueStatement / If(Expression){Continue;}
    3. General Branch Introduction
13. Initializer Introduction

**Expression transformation**

Expression transformation should consider 1) the context syntactic structure and 2) information coming from fault localization.

1. Decision Blocks

Decision blocks may be in the context of Branch and Loops:

1. IfStatement: If(Expression) {} [else{}]
2. SwitchStatement + SwitchCase=> switch(Expression1){ case(Expression2){}} -> Expression1== Expression2
3. DoStatement: do{}while(Expression)
4. ForStatement: for(InitExpression; DecisionExpression; UpdateExpression){} -> DecisionExpression
5. WhileStatement: while(Expression){}

These Boolean expressions would be transformed in the following ways:

1. Expression || ?
2. Expression && ?
3. !Expression
4. General Expression transformations
5. Execution Blocks

If we located statements that may affect the concerned variable, then the statement can be transformed according to their syntactical structure:

1. ConstructorInvocation
   1. Change to Super
   2. Change to Child
   3. Change to another constructor
   4. Modify passed parameters as expressions
2. ExpressionStatement
   1. Assignment Expression as expression
3. LabeledStatement
   1. General statement
4. ReturnStatement
   1. Change return expression
      1. Find the operator that affects the concerned variable and modify the corresponding expression
      2. Return different value
5. SuperConstructorInvocation
   1. Change to another constructor
   2. Modify passed parameters as expressions
6. VariableDeclarationStatement
   1. If the variable is initialized, modify the initialization Expression

For general expressions:

1. Constant
2. Method Call
3. NullChecker
4. CastChecker