

1 Renaming Handler

1.1 Early Concepts

1.1.1 Possibly renamed without body changes nodes

$$R_{wobc}(T, B) = \{b \in B \mid (\neg \exists t \in T (t.id = b.id)) \wedge (\exists t \in T (t.body = b.body))\}$$

1.1.2 Possibly deleted or renamed with body changes nodes

$$DR_{wbc}(T, B) = \{b \in B \mid \neg \exists t \in T (t.id = b.id \vee t.body = b.body)\}$$

1.2 Match Algorithm

Algorithm 1: Match Algorithm

```
Input: L, B, R, M
Output: Set of quadruples  $(l, b, r, m)$  consisting of the base node  $b$  and its corresponding left
node  $l$ , right node  $r$  and merge node  $m$ 

1  $matches \leftarrow \emptyset$ ;
2 foreach  $b$  in  $possiblyRenamedOrDeletedBaseNodes(L, B, R)$  do
3    $l \leftarrow getCorrespondentNode(b, L)$ ;
4    $r \leftarrow getCorrespondentNode(b, R)$ ;
5    $m \leftarrow getMergeNode(l, r, M)$ ;
6    $matches \leftarrow matches \cup (l, b, r, m)$ ;
7 end
8 return  $matches$ 
9 function  $possiblyRenamedOrDeletedBaseNodes(L, B, R)$ 
10 | return  $DR_{wbc}(L, B) \cup DR_{wbc}(R, B) \cup R_{wobc}(L, B) \cup R_{wobc}(R, B)$ ;
11 end
12 function  $getCorrespondentNode(b, T)$ 
13 |  $t \leftarrow findFirst(t \in T \rightarrow t.id = b.id)$ ;
14 | if  $t = null$  then
15 | |  $t \leftarrow findFirst(t \in T \rightarrow t.body = b.body)$ ;
16 | end
17 | if  $t = null$  then
18 | |  $t \leftarrow findFirst(t \in T \rightarrow t.body \approx b.body \wedge (t.id.name = b.id.name \vee$ 
19 | |  $t.id.params = b.id.params))$ ;
20 | end
21 | if  $t = null$  then
22 | |  $t \leftarrow findFirst(t \in T \rightarrow t.body = substring(b.body) \vee b.body = substring(t.body))$ ;
23 | end
24 | return  $t$ ;
25 function  $getMergeNode(l, r, M)$ 
26 | if  $l \neq null$  then
27 | | return  $find(m \in M \rightarrow m.id = l.id)$ ;
28 | end
29 | if  $r \neq null$  then
30 | | return  $find(m \in M \rightarrow m.id = r.id)$ ;
31 | end
32 | return  $null$ ;
33 end
```

1.3 Handle Algorithms

Algorithm 2: Check References and Merge Methods Variant

```

Input:  $(l, b, r, m), M$ 
1 if  $\text{singleRenamingOrDeletion}(l, b, r)$  then
2    $m.body = \text{textualMerge}(l, b, r);$ 
3    $\text{removeUnmatchedNode}(l, r, m, M);$ 
4 else if  $l.id \neq r.id$  then
5    $m.body = \text{conflict}(l, b, r);$ 
6    $\text{removeUnmatchedNode}(l, r, m, M);$ 
7 else if  $l.body \neq r.body$  then
8   if  $\text{newReferenceTo}(l) \vee \text{newReferenceTo}(r)$  then
9      $m.body = \text{conflict}(l, b, r);$ 
10     $\text{removeUnmatchedNode}(l, r, m, M);$ 
11   else
12      $m.body = \text{textualMerge}(l, b, r);$ 
13      $\text{removeUnmatchedNode}(l, r, m, M);$ 
14   end
15 end
16 function  $\text{singleRenamingOrDeletion}(l, b, r)$ 
17   return  $l.id = b.id \vee r.id = b.id;$ 
18 end
19 function  $\text{removeUnmatchedNode}(l, r, m, M)$ 
20   if  $l.id = m.id \wedge r.id \neq m.id$  then
21      $\text{removeNode}(r, M);$ 
22   end
23 end

```

Algorithm 3: Merge Methods Variant

```

Input:  $(l, b, r, m), M$ 
1  $m.body = \text{textualMerge}(l, b, r);$ 
2  $\text{removeUnmatchedNode}(l, r, m, M);$ 
3 function  $\text{removeUnmatchedNode}(l, r, m, M)$ 
4   if  $l.id = m.id \wedge r.id \neq m.id$  then
5      $\text{removeNode}(r, M);$ 
6   end
7 end

```

Algorithm 4: Check Textual and Keep Both Methods Variant**Input:** $(l, b, r, m), M$

```

1 if singleRenamingOrDeletion( $l, b, r$ ) then
2   if textualMergeHasConflictInvolvingSignature( $b$ ) then
3      $m.body = conflict(l, b, r);$ 
4      $removeUnmatchedNode(l, r, m, M);$ 
5   end
6 else if  $l.id \neq r.id \wedge l.body = r.body$  then
7    $m.body = conflict(l, b, r);$ 
8    $removeUnmatchedNode(l, r, m, M);$ 
9 end
10 function singleRenamingOrDeletion( $l, b, r$ )
11   return  $l.id = b.id \vee r.id = b.id;$ 
12 end
13 function removeUnmatchedNode( $l, r, m, M$ )
14   if  $l.id = m.id \wedge r.id \neq m.id$  then
15      $removeNode(r, M);$ 
16   end
17 end

```

Algorithm 5: Keep Both Methods Variant**Input:** $(l, b, r, m), M$

```

1 if singleRenamingOrDeletion( $l, b, r$ )  $\wedge$  hasConflict( $m$ ) then
2    $removeConflict(m);$ 
3 end
4 function singleRenamingOrDeletion( $l, b, r$ )
5   return  $l.id = b.id \vee r.id = b.id;$ 
6 end

```