0.1 New Element Referencing Edited One Handler

0.1.1 Handler Algorithm

Algorithm 1: Handle

Input: L, B, R, M

$M_U \leftarrow$ `textualMerge(treeToText(`L`), treeToText(`B`), treeToText(`R`))`;
$cs \leftarrow$ `extractConflicts(`M_U`)`;
$aMFD_L \leftarrow \{l \in A_L \mid l.type = METHODDECL \vee l.type = FIELDDECL\}$;
$aMFD_R \leftarrow \{r \in A_R \mid r.type = METHODDECL \vee r.type = FIELDDECL\}$;
$eMFD_L \leftarrow \{l \in E_L \mid l.type = METHODDECL \vee l.type = FIELDDECL\}$;
$eMFD_R \leftarrow \{r \in E_R \mid r.type = METHODDECL \vee r.type = FIELDDECL\}$;
foreach $a_l \in aMFD_L$ **do**
  foreach $e_r \in eMFD_R$ **do**
    if `nodesConflict(`a_l`,` e_r`,` cs`)` $\wedge\ e_r.id.name \in a_l.body$ **then**
      $b \leftarrow find(b \in B \rightarrow b.id = e_r.id)$;
      $m \leftarrow find(m \in M \rightarrow m.body = e_r.body)$;
      $m.body \leftarrow conflict(e_r.body, b.body, a_l.body)$;
      $m \leftarrow find(m \in M \rightarrow m.body = a_l.body)$;
      `removeNode(`m`,` M`)`;
    end
  end
end
foreach $a_r \in aMFD_R$ **do**
  foreach $e_l \in eMFD_L$ **do**
    if `nodesConflict(`a_r`,` e_l`,` cs`)` $\wedge\ r_l.id.name \in a_r.body$ **then**
      $b \leftarrow find(b \in B \rightarrow b.id = e_l.id)$;
      $m \leftarrow find(m \in M \rightarrow m.body = e_l.body)$;
      $m.body \leftarrow conflict(e_l.body, b.body, a_r.body)$;
      $m \leftarrow find(m \in M \rightarrow m.body = a_r.body)$;
      `removeNode(`m`,` M`)`;
    end
  end
end

Algorithm 2: Nodes Conflict

Input: a, b, cs
Output: wether there is an unstructured conflict in cs concerning a and b nodes

foreach $c \in cs$ **do**
  if $c.left = a.body \wedge c.right = b.body$ **then return** $TRUE$;
  if $c.left = b.body \wedge c.right = a.body$ **then return** $TRUE$;
end
return $FALSE$;