In Algorithm 1, the linguistic conformance check of the MvK is shown. It checks whether a given model conforms to a given type model. It consists of four checks:

1. (Lines 1-7) Checks whether all elements in the model are typed by an element in the type model. Functions used are:
   (a) `populate_types`: returns a mapping between types of the type model, and instances of those type in the model.
   (b) `type`: returns the type of an element.
2. (Lines 8-14) Checks whether the minimum and maximum cardinality for each type in the type model is satisfied. Functions used are:
   (a) `get_minimum`: returns the minimum cardinality for an given type.
   (b) `get_maximum`: returns the maximum cardinality for an given type.
3. (Lines 15-24) Checks, for all attributes of all elements in the model, whether a type definition for the attribute can be found in the type model (lines 17-19), and whether the type of the attribute value corresponds to the type defined in the attribute type (lines 20-22). Functions used are:
   (a) `get_attributes`: returns all attributes of an element.
   (b) `get_value`: returns the value of an attribute.
   (c) `get_type`: returns the type of the values which can be assigned to an attribute.
4. (Lines 25-49) Checks, for each incoming and outgoing association of each element of the model, whether, respectively, the incoming and outgoing cardinalities are satisfied. Also checks whether the types of the connected elements correspond to those defined in the association. Functions used are:
   (a) `classify_by_type`: classifies the given elements by their type, and returns a mapping between types and instances.
   (b) `get_outAssociations`: returns all outgoing associations of an element.
   (c) `get_inAssociations`: returns all incoming associations of an element.
   (d) `get_minimum_out`: returns the minimum number of outgoing associations of a particular type.
Algorithm 1 The MvK’s linguistic conformance check.

**Input:** model, type_model

1: type_to_elements ← populate_types(model, type_model)
2: for el in model do
3:   if not type(el) in type_to_elements then
4:     return False
5:   end if
6:   append el to type_to_elements[type(el)]
7: end for
8: for type in type_to_elements do
9:   if len(type_to_elements[type]) < get_minimum(type) then
10:      return False
11:   end if
12:   if len(type_to_elements[type]) > get_maximum(type) then
13:      return False
14:   end if
15: for el in type_to_elements[type] do
16:   for attr in get_attributes(el) do
17:     if not type(attr) in get_attributes(type) then
18:       return False
19:     end if
20:     if not type(get_value(attr)) = get_type(type(attr)) then
21:       return False
22:     end if
23:   end for
24: end for
25: out_associations ← classify_by_type(get_out_associations(el))
26: for assoc_type in out_associations do
27:   if not(get_in_type(assoc_type) in get_all_types(el)) then
28:     return False
29:   end if
30:   if len(out_associations[assoc_type]) < get_minimum_out(assoc_type) then
31:     return False
32:   end if
33:   if len(out_associations[assoc_type]) > get_maximum_out(assoc_type) then
34:     return False
35:   end if
36: end for
37: in_associations ← classify_by_type(get_in_associations(el))
38: for assoc_type in in_associations do
39:   if not(get_out_type(assoc_type) in get_all_types(el)) then
40:     return False
41:   end if
42:   if len(in_associations[assoc_type]) < get_minimum_in(assoc_type) then
43:     return False
44:   end if
45:   if len(in_associations[assoc_type]) > get_maximum_in(assoc_type) then
46:     return False
47:   end if
48: end for
49: end for
50: return True
(e) `get_maximum_out`: returns the maximum number of outgoing associations of a particular type.
(f) `get_minimum_in`: returns the minimum number of incoming associations of a particular type.
(g) `get_maximum_in`: returns the maximum number of incoming associations of a particular type.
(h) `get_out_type`: returns the type defined for the outgoing multiplicity of an association.
(i) `get_in_type`: returns the type defined for the incoming multiplicity of an association.
(j) `get_all_types`: returns the type of the element, as well as all its subtypes.