ALGORITHM 1: NCKG MODELING

```
Input: C ← contract clause,
       N ← ontology of NCKG
Initialize: ContractActorClass← [],  ContractObjectClass← [], EventClass ←
[], StatementClass ← [], ConstraintClass ← [], PropertyClass ← []

1  for clause in C
2  do
3      (actor, relation, object) ← matchAllInstance(Contract_actor.subclass,
       hasActionTo.subclass, Contract_object.subclass);
4      if (actor, relation, object)=None then
5          ContractActorClass.append(actor)
6          ContractObjectClass.append(object)
7      end
8      else
9          event ←SetFact(actor, relation, object)
10         EventClass.append(event)
11     end
12     (object, relation, property) ←matchAllInstance(Contract_object.subclass,
       Property.subclass)
13     if (object, relation, property)=None then
14         PropertyClass.append(property)
15     end
16     else
17         statement ←SetFact(object, relation, property)
18         StatementClass.append( statement )
19     end
20     constraint ← matchAllInstance(Constraint.subclass)
21     constraint ←SetEntity(constraint)
22     ConstraintClass.append(constraint)
23     for evt, stat, constr in zip(EventClass, StatementClass, ConstraintClass) do
24         (evt, hasConstraint, constr) ←linkRelation(evt, constr)
25         (evt, hasContractualRelation, evt) ←linkRelation(evt, evt)
26         (evt, hasContractualRelation, stat) ←linkRelation(evt, stat)
27     end
28     wrap all representations in RDF-star
29 end
```