Abdul Moiz Nazir

It's basic structure, to extend and modify it according to the specific details of OOASP model and requirements. Also, including rules for handling relationships, constraints, and any additional features described in OOASP table. It's necessary to have separate files for config-model.lp and partial-instantiation.lp to keep your code organized.

```
% ooasp.lp

% Define classes, attributes, and relationships
class(person).
class(address).

attribute(person, name).
attribute(person, age).
attribute(address, city).

relationship(has_address, person, address).

% Define instantiation rules based on partial instantiation
instantiated(name, "John"). % Example partial instantiation
instantiated(age, 25).

% Rules to complete the instantiation
instantiated(Attribute, Value) :-
    attribute(Class, Attribute),
    class(Class),
    not instantiated(Attribute, _),
    default_value(Attribute, DefaultValue),
    not instantiated(Attribute, DefaultValue),
    assert(instantiated(Attribute, DefaultValue)).

% Define output format predicates (adjust as per Table 2)
output_predicate(Attribute, Value) :-
    instantiated(Attribute, Value).

% Additional rules and constraints as needed
```

In this example, I added an integrity constraint to check the minimum cardinality of associations. It’s needed to extend these integrity constraints based on the specific constraints mentioned in the configuration model.

```
% ooasp.lp
```

```
% Define classes, attributes, and relationships
class(person).
class(address).

attribute(person, name).
attribute(person, age).
attribute(address, city).

relationship(has_address, person, address).

% Define instantiation rules based on partial instantiation
instantiated(name, "John"). % Example partial instantiation
instantiated(age, 25).

% Rules to complete the instantiation
instantiated(Attribute, Value) :-
    attribute(Class, Attribute),
    class(Class),
    not instantiated(Attribute, _),
    default_value(Attribute, DefaultValue),
    not instantiated(Attribute, DefaultValue),
    assert(instantiated(Attribute, DefaultValue)).

% Define integrity constraints for minimum association cardinality
:- relationship(Relation, Class1, Class2),
   min_cardinality(Relation, MinCard),
   count_associations(Relation, Class1, Class2, Count),
   Count < MinCard.

% Rules to count associations
count_associations(Relation, Class1, Class2, Count) :-
    findall(_, has_association(Relation, Class1, Class2), Associations),
    length(Associations, Count).

% Define output format predicates (adjust as per Table 2)
output_predicate(Attribute, Value) :-
    instantiated(Attribute, Value).

% Additional rules and constraints as needed
```

Test case for the Modules configuration model in Listing 1 along with the partial instantiation provided in subsection 4.2.

```
% config-model.lp

% Classes
ooasp_class("v1", "HwObject").
ooasp_class("v1", "Frame").
ooasp_class("v1", "Module").
ooasp_class("v1", "ModuleA").
ooasp_class("v1", "ModuleB").
ooasp_class("v1", "Element").
ooasp_class("v1", "ElementA").
ooasp_class("v1", "ElementB").

% Class inheritance
ooasp_subclass("v1", "Frame", "HwObject").
ooasp_subclass("v1", "Module", "HwObject").
ooasp_subclass("v1", "Element", "HwObject").
ooasp_subclass("v1", "ElementA", "Element").
ooasp_subclass("v1", "ElementB", "Element").
ooasp_subclass("v1", "ModuleA", "Module").
ooasp_subclass("v1", "ModuleB", "Module").

% Attributes and associations
% Class Frame
ooasp_assoc("v1", "Frame_modules", "Frame", 1, 1, "Module", 0, 5).

% Class Module
ooasp_attribute("v1", "Module", "position", "integer").
ooasp_attribute_minInclusive("v1", "Module", "position", 1).
ooasp_attribute_maxInclusive("v1", "Module", "position", 5).

% Class Element
ooasp_assoc("v1", "Element_module", "Element", 1, 1, "Module", 1, 1).
```

Created a partial instantiation for testing:

```
% partial-instantiation.lp

% Partial instantiation for testing
instantiated ("position", 3).
```

Now we can use main encoding (ooasp.lp) and check if it generates the complete instantiation correctly while satisfying the constraints defined in the configuration model.

The command provided seems correct for using Clingo to generate instantiations. The -0 option is used to disable optimization, which is suitable for finding models without any optimization criteria.

clingo config-model.lp partial-instantiation.lp ooasp.lp -0

After installing Clingo on the system, the files `config-model.lp`, `partial-instantiation.lp`, and `ooasp.lp` are in the same directory or provide the correct paths.

5. With the minimal number of components, we can use the Clingo optimization feature. It's extended to the encoding to include optimization:

```
% ooasp.lp

% ... (previous code)

% Define optimization statement
#minimize {1, Attribute, Value: instantiated (Attribute, Value)}.

% Additional rules and constraints as needed
```

This #minimize statement aims to minimize the number of instantiated attributes in the solution, effectively finding the instantiation with the minimal number of components.