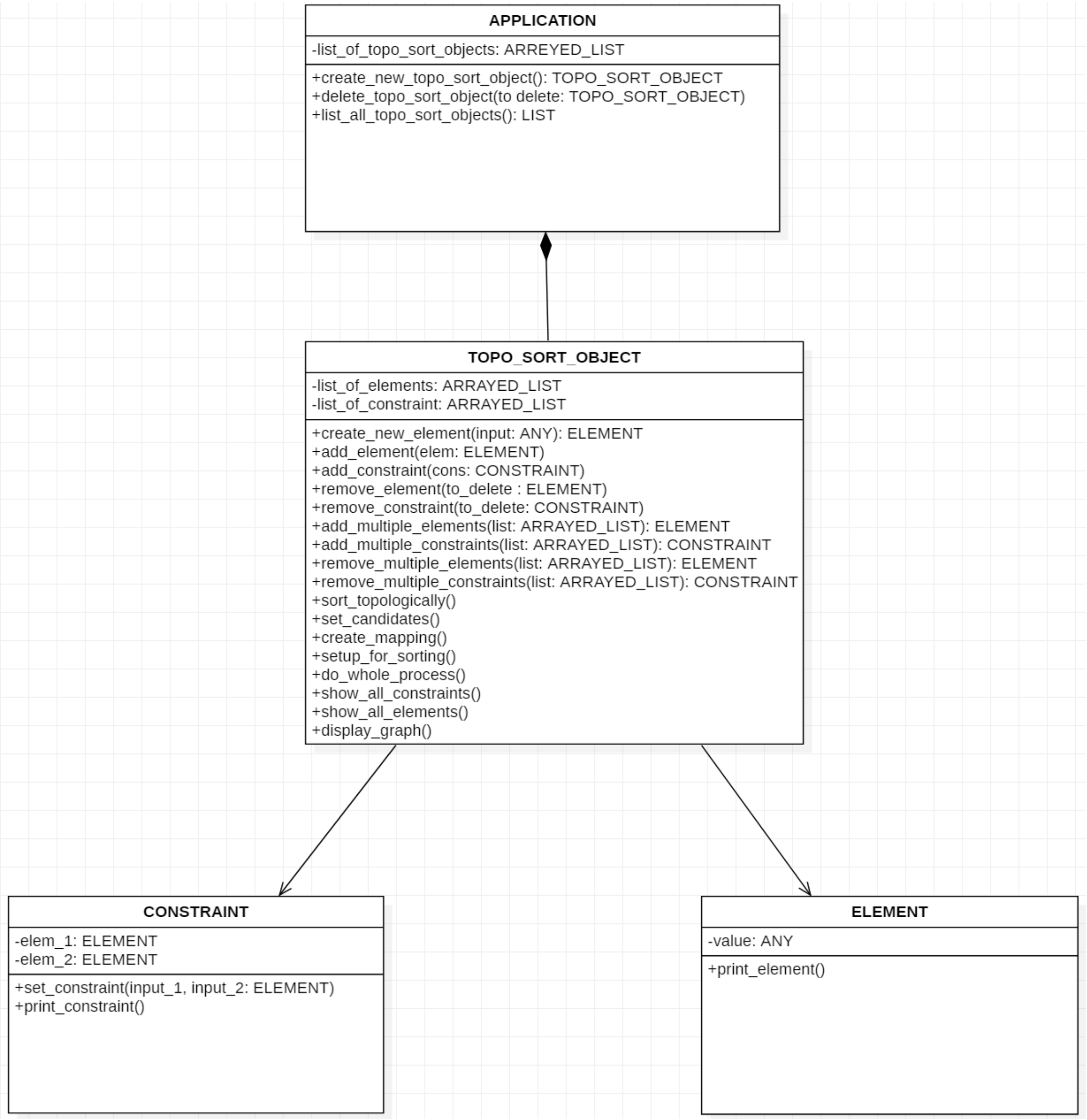
Design document

UML


APPLICATION
-list_of_topo_sort_objects: ARREYED_LIST
+create_new_topo_sort_object(): TOPO_SORT_OBJECT
+delete_topo_sort_object(to delete: TOPO_SORT_OBJECT)
+list_all_topo_sort_objects(): LIST
TOPO_SORT_OBJECT
-list_of_elements: ARRAYED_LIST
-list_of_constraint: ARRAYED_LIST
+create_new_element(input: ANY): ELEMENT
+add_element(elem: ELEMENT)
+add_constraint(cons: CONSTRAINT)
+remove_element(to_delete : ELEMENT)
+remove_constraint(to_delete: CONSTRAINT)
+add_multiple_elements(list: ARRAYED_LIST): ELEMENT
+add_multiple_constraints(list: ARRAYED_LIST): CONSTRAINT
+remove_multiple_elements(list: ARRAYED_LIST): ELEMENT
+remove_multiple_constraints(list: ARRAYED_LIST): CONSTRAINT
+sort_topologically()
+set_candidates()
+create_mapping()
+setup_for_sorting()
+do_whole_process()
+show_all_constraints()
+show_all_elements()
+display_graph()
CONSTRAINT
-elem_1: ELEMENT
-elem_2: ELEMENT
+set_constraint(input_1, input_2: ELEMENT)
+print_constraint()
ELEMENT
-value: ANY
+print_element()

Design Pattern: Factory Pattern

In our Eiffel library we used the four following classes: APPLICATION, TOPO_SORT_OBJECT, CONSTRAINT, ELEMENT. APPLICATION is our root class and inherits from no other class. TOPO_SORT_OBJECT is the class which does most of the work. In this class the user adds and removes elements and constraints. The user can do this with single elements or constraints but also with a list of multiple elements or constraints. Within this class is the operation sort_topologically() defined which sorts the elements according to the constraints in topological order. This method sorts them without any cycles. After the elements are sorted in topological order the user can display a graph of the element in the correct order with the operation display_graph(). CONSTRAINT is the class we use to create the different constraints which are then stored in the list_of_constraints. ELEMENT is the class we use to create elements which are then stored in the list_of_elements.

We used the factory design pattern for our Eiffel library. This pattern allows us to have different TOPO_SORT_OBJECT. A user can call the APPLICATION to create a new TOPO_SORT_OBJECT. This enables the user to have multiple TOPO_SORT_OBJECTs at the same time and he can work on them simultaneously. The TOPO_SORT_OBJECT does the main work and uses the classes ELEMENT and CONSTRAINT to work with these two within its own class.