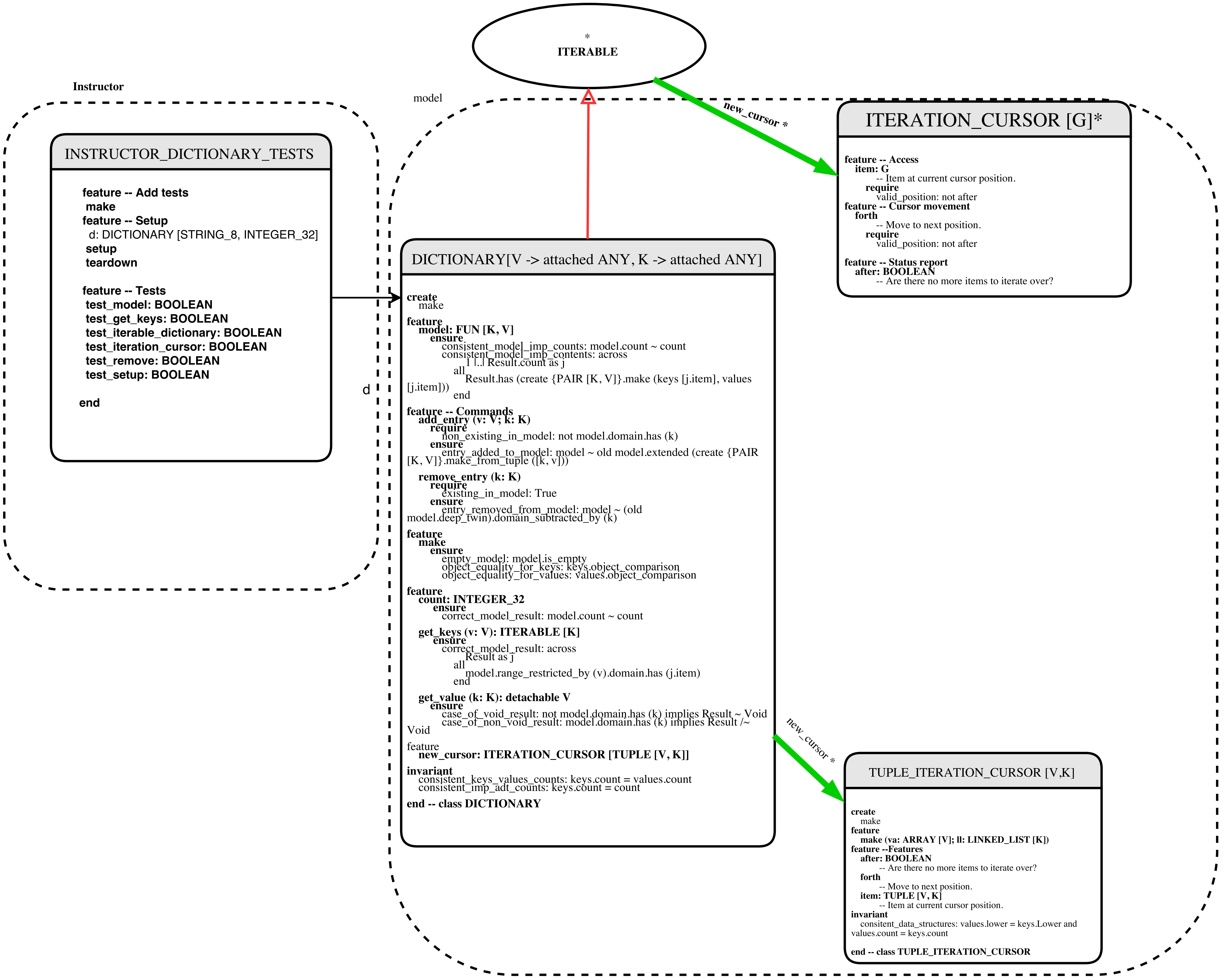

```
INSTRUCTOR_DICTIONARY_TESTS

  feature -- Add tests
   make
  feature -- Setup
    d: DICTIONARY [STRING_8, INTEGER_32]
   setup
   teardown

  feature -- Tests
   test_model: BOOLEAN
   test_get_keys: BOOLEAN
   test_iterable_dictionary: BOOLEAN
   test_iteration_cursor: BOOLEAN
   test_remove: BOOLEAN
   test_setup: BOOLEAN

  end
```

```
DICTIONARY[V -> attached ANY, K -> attached ANY]

create
    make
feature
    model: FUN [K, V]
        ensure
            consistent_model_imp_counts: model.count ~ count
            consistent_model_imp_contents: across
                1 |..| Result.count as j
            all
                Result.has (create {PAIR [K, V]}.make (keys [j.item], values
[j.item]))
            end
feature -- Commands
    add_entry (v: V; k: K)
        require
            non_existing_in_model: not model.domain.has (k)
        ensure
            entry_added_to_model: model ~ old model.extended (create {PAIR
[K, V]}.make_from_tuple ([k, v]))
    remove_entry (k: K)
        require
            existing_in_model: True
        ensure
            entry_removed_from_model: model ~ (old
model.deep_twin).domain_subtracted_by (k)
feature
    make
        ensure
            empty_model: model.is_empty
            object_equality_for_keys: keys.object_comparison
            object_equality_for_values: values.object_comparison
feature
    count: INTEGER_32
        ensure
            correct_model_result: model.count ~ count
    get_keys (v: V): ITERABLE [K]
        ensure
            correct_model_result: across
                Result as j
            all
                model.range_restricted_by (v).domain.has (j.item)
            end
    get_value (k: K): detachable V
        ensure
            case_of_void_result: not model.domain.has (k) implies Result ~ Void
            case_of_non_void_result: model.domain.has (k) implies Result /~
Void
feature
    new_cursor: ITERATION_CURSOR [TUPLE [V, K]]
invariant
    consistent_keys_values_counts: keys.count = values.count
    consistent_imp_adt_counts: keys.count = count
end -- class DICTIONARY
```

```
ITERATION_CURSOR [G]*

feature -- Access
   item: G
         -- Item at current cursor position.
      require
         valid_position: not after
feature -- Cursor movement
   forth
         -- Move to next position.
      require
         valid_position: not after

feature -- Status report
   after: BOOLEAN
         -- Are there no more items to iterate over?
```

```
TUPLE_ITERATION_CURSOR [V,K]

create
   make
feature
   make (va: ARRAY [V]; ll: LINKED_LIST [K])
feature --Features
   after: BOOLEAN
         -- Are there no more items to iterate over?
   forth
         -- Move to next position.
   item: TUPLE [V, K]
         -- Item at current cursor position.
invariant
   consitent_data_structures: values.lower = keys.Lower and
values.count = keys.count

end -- class TUPLE_ITERATION_CURSOR
```