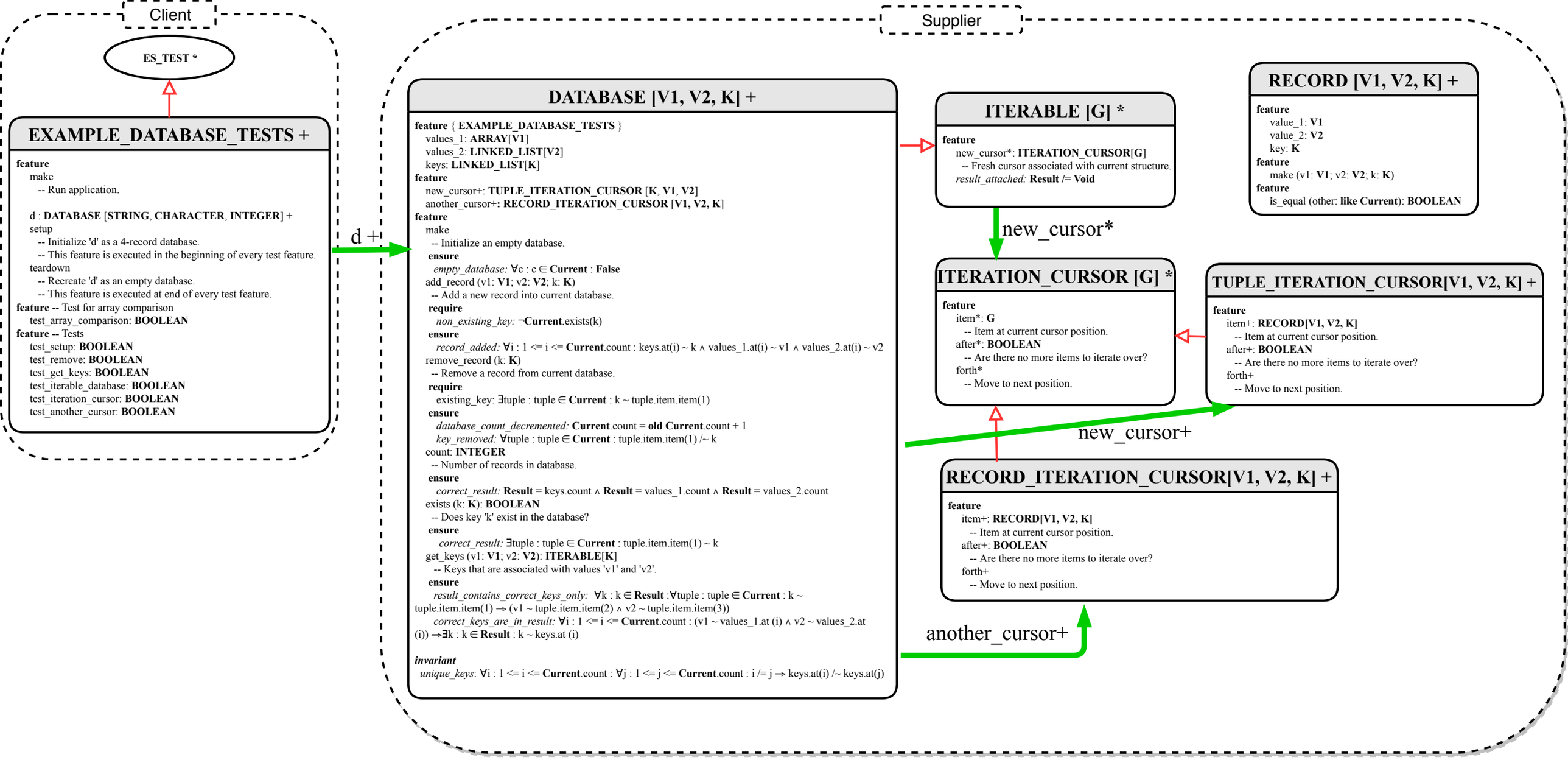
Client
ES_TEST *
EXAMPLE_DATABASE_TESTS +
feature
make
-- Run application.
d : DATABASE [STRING, CHARACTER, INTEGER] +
setup
-- Initialize 'd' as a 4-record database.
-- This feature is executed in the beginning of every test feature.
teardown
-- Recreate 'd' as an empty database.
-- This feature is executed at end of every test feature.
feature -- Test for array comparison
test_array_comparison: BOOLEAN
feature -- Tests
test_setup: BOOLEAN
test_remove: BOOLEAN
test_get_keys: BOOLEAN
test_iterable_database: BOOLEAN
test_iteration_cursor: BOOLEAN
test_another_cursor: BOOLEAN
d +
Supplier
DATABASE [V1, V2, K] +
feature { EXAMPLE_DATABASE_TESTS }
values_1: ARRAY[V1]
values_2: LINKED_LIST[V2]
keys: LINKED_LIST[K]
feature
new_cursor+: TUPLE_ITERATION_CURSOR [K, V1, V2]
another_cursor+: RECORD_ITERATION_CURSOR [V1, V2, K]
feature
make
-- Initialize an empty database.
ensure
empty_database: ∀c : c ∈ Current : False
add_record (v1: V1; v2: V2; k: K)
-- Add a new record into current database.
require
non_existing_key: ¬Current.exists(k)
ensure
record_added: ∀i : 1 <= i <= Current.count : keys.at(i) ~ k ∧ values_1.at(i) ~ v1 ∧ values_2.at(i) ~ v2
remove_record (k: K)
-- Remove a record from current database.
require
existing_key: ∃tuple : tuple ∈ Current : k ~ tuple.item.item(1)
ensure
database_count_decremented: Current.count = old Current.count + 1
key_removed: ∀tuple : tuple ∈ Current : tuple.item.item(1) /~ k
count: INTEGER
-- Number of records in database.
ensure
correct_result: Result = keys.count ∧ Result = values_1.count ∧ Result = values_2.count
exists (k: K): BOOLEAN
-- Does key 'k' exist in the database?
ensure
correct_result: ∃tuple : tuple ∈ Current : tuple.item.item(1) ~ k
get_keys (v1: V1; v2: V2): ITERABLE[K]
-- Keys that are associated with values 'v1' and 'v2'.
ensure
result_contains_correct_keys_only: ∀k : k ∈ Result :∀tuple : tuple ∈ Current : k ~ tuple.item.item(1) ⇒ (v1 ~ tuple.item.item(2) ∧ v2 ~ tuple.item.item(3))
correct_keys_are_in_result: ∀i : 1 <= i <= Current.count : (v1 ~ values_1.at (i) ∧ v2 ~ values_2.at (i)) ⇒∃k : k ∈ Result : k ~ keys.at (i)
invariant
unique_keys: ∀i : 1 <= i <= Current.count : ∀j : 1 <= j <= Current.count : i /= j ⇒ keys.at(i) /~ keys.at(j)
ITERABLE [G] *
feature
new_cursor*: ITERATION_CURSOR[G]
-- Fresh cursor associated with current structure.
result_attached: Result /= Void
new_cursor*
RECORD [V1, V2, K] +
feature
value_1: V1
value_2: V2
key: K
feature
make (v1: V1; v2: V2; k: K)
feature
is_equal (other: like Current): BOOLEAN
ITERATION_CURSOR [G] *
feature
item*: G
-- Item at current cursor position.
after*: BOOLEAN
-- Are there no more items to iterate over?
forth*
-- Move to next position.
TUPLE_ITERATION_CURSOR[V1, V2, K] +
feature
item+: RECORD[V1, V2, K]
-- Item at current cursor position.
after+: BOOLEAN
-- Are there no more items to iterate over?
forth+
-- Move to next position.
new_cursor+
RECORD_ITERATION_CURSOR[V1, V2, K] +
feature
item+: RECORD[V1, V2, K]
-- Item at current cursor position.
after+: BOOLEAN
-- Are there no more items to iterate over?
forth+
-- Move to next position.
another_cursor+