EECS3311 – Software Design

Section A, Fall 2019

Lab 2 – Implementing an Iterable Repository ADT – Report

Umar Abdulselam

uaabduls

215995616

PART 1: Explain how the Iterator Pattern is implemented in the model cluster

The model cluster is comprised of four components: REPOSITORY, DATA_SET, TUPLE_ITERATION_CURSOR, and DATA_SET_ITERATION_CURSOR. A data set is a set of data that can be accessed by those who have access to it. The DATA_SET class is a definition of a data set object containing three components: a key and two values.

A repository is an iterable abstract data type allowing for storage of objects containing a key and its associated values. The REPOSITORY class is an implementation of said abstract data type, allowing for storage of objects containing a key and two data values. That is, a REPOSITORY is an iterable object made up of collections of values from DATA_SET objects. In the current implementation of a REPOSITORY, a LINKED_LIST of keys, an ARRAY of value1 and a HASH_TABLE of (key, value2) pairs are maintained.

In REPOSITORY, the iterator pattern is implemented, enabling the information hiding principle. The REPOSITORY is iterable with its underlying data structures exported to only a test suite. This allows any client of REPOSITORY to not be concerned about the underlying implementation of REPOSITORY, but rather just expect that the REPOSITORY is iterable.

Since a REPOSITORY consists of multiple (three) iterable data structures, it is not possible to iterate the REPOSITORY in unison using the cursors of each respective underlying data structure. Hence, the TUPLE_ITERATION_CURSOR and DATA_SET_ITERATION_CURSOR objects are defined.

The TUPLE_ITERATION_CURSOR and DATA_SET_ITERATION_CURSOR objects facilitate the iteration of a REPOSITORY's data structures in unison. Either of these cursors allow a client of REPOSITORY to iterate a REPOSITORY object and access the {key, value1, value2} sets in the underlying data structures regardless of the fact that the DATA_ITEM components are stored in separate data structures. This access is facilitated by use of the TUPLE_ITERATION_CURSOR and DATA_SET_ITERATION_CURSOR's features: *cursor_position*, *item*, *after*, and *forth*.

As depicted in the BON diagram, a REPOSITORY inherits from ITERABLE and is the client of a TUPLE_ITERATION_CURSOR and a DATA_SET_ITERATION_CURSOR, both of whom inherit from ITERATION_CURSOR. In the model cluster, all deferred features inherited by descendant classes are implemented to be effective. Namely, REPOSITORY implements *new_cursor* deferred from ITERABLE, and TUPLE_ITERATION_CURSOR and DATA_SET_ITERATION_CURSOR implement *item*, *after*, and *forth*, deferred from ITERATION_CURSOR.

PART 2: Explain how you implement the feature *another_cursor* in the REPOSITORY class.

The feature '*another_cursor*' is implemented in a similar way to the '*new_cursor*' feature deferred from the ITERABLE class. The difference between TUPLE_ITERATION_CURSOR and DATA_SET_ITERATION_CURSOR is that the former returns a cursor object of type TUPLE, while the latter returns a cursor object of type DATA_SET. Both of the iteration cursors return objects that can be accessed by position (fields) to get the corresponding key, value1 or value2 item in a REPOSITORY when iterated.

Similar to '*new_cursor*', '*another_cursor*' returns a DATA_SET_ITERATION_CURSOR that implements deferred features *item*, *after*, and *forth*. Feature *item* returns the key and two values associated with the DATA_SET object at position '*cursor_position'*. In the case of DATA_SET_ITERATION_CURSOR, it must be checked that value2 being assigned to the cursor is attached since the actual type of the field is attached and hence cannot be deferred. Features *after* and *forth* are implemented in the same way as they are in TUPLE_ITERATION_CURSOR.