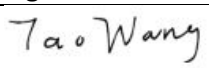
Tracker Project

Team members	Name	Prism Login	Signature
Member 1:	Tao Wang	kevin89	
*Submitted under Prism account: kevin89			

Contents

1. Requirements for Tracker Project

A tracker system monitors the position of waste products in nuclear plants and ensures their safe handling. Our customer requires a software system that operators use to manage safe tracking of radioactive waste in their various nuclear plants.

We have so far elicited the following information from our customer.

Containers of material pass through various stages of processing in the tracking part of the nuclear plant. The tracking plant consist of several phases usually corresponding to the physical processes that handle the radioactive materials. Not all plants have precisely the same phases.

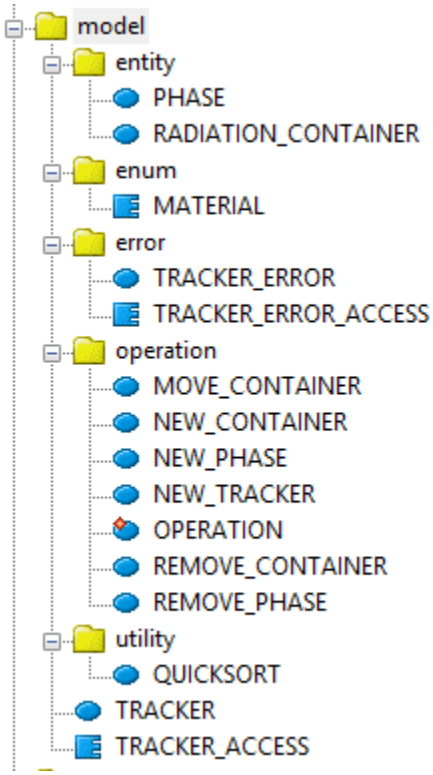
For a given plant, there is an initial setup of two important fixed global parameters: there is a limit on the maximum radiation in any phase of the plant , and there is also a limit on the maximum radiation that any container in the plant may have. An error status message shall be signaled if there is an attempt to register a new container in the system with radiation that exceeds the container limit.

Another operation is to add a new phase (this is information provided by the Domain experts). Requirements elicitation so far yields that a new phase is specified by a phase ID, a name (e.g. "compacting"), a limit on the maximum number of containers in the phase, and a list of material types that may be treated in the phase. A phase may also be removed if there are no containers anywhere in the system. Also, it is possible for an operator to move a container from one phase to another.

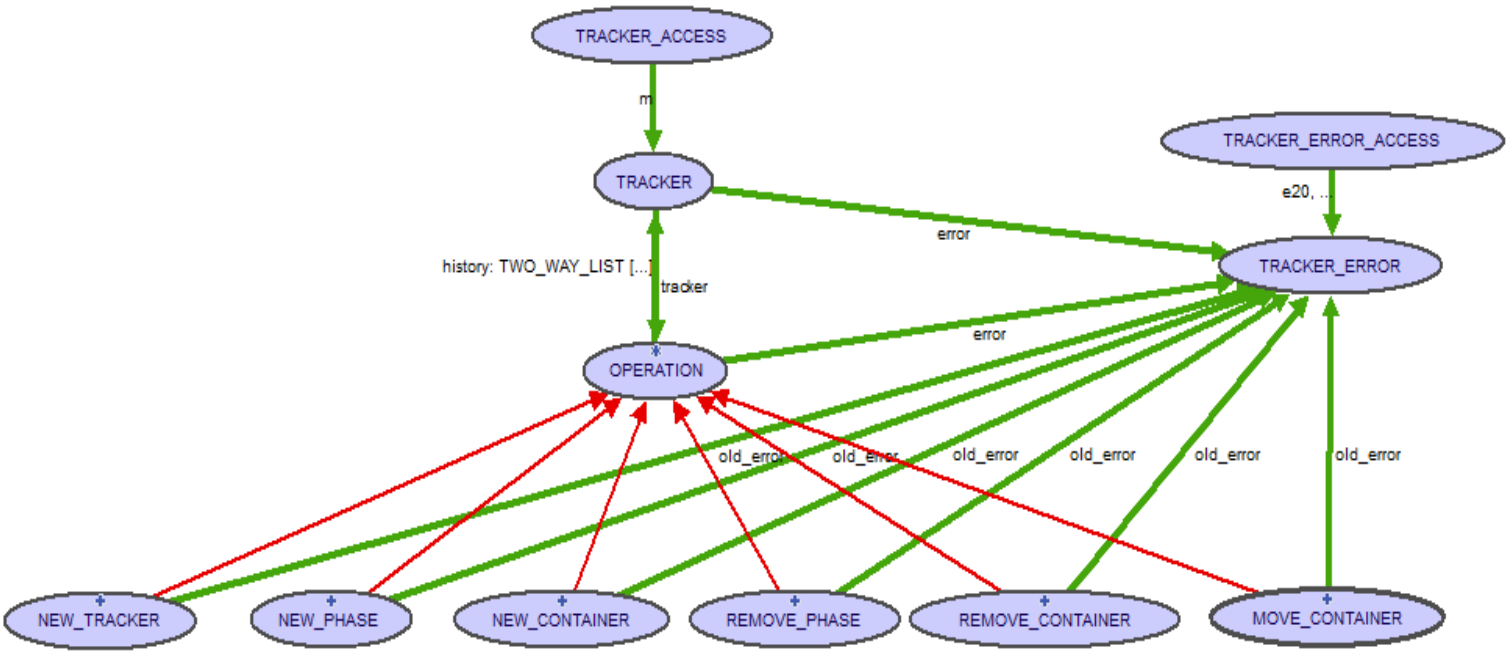
Obviously when dealing with dangerous materials, it is very important to ensure that no material goes missing and that care is taken to avoid too much radioactive material getting into a phase, in case there is a buildup of dangerous substances in one area. The tracking manager is responsible for giving permission to movements of containers between processing phases to avoid dangerous situations.

2. BON class diagram overview (architecture of the design)

The file structure in the cluster model is as follows



The main BON diagram is below



Based on the idea of singleton, there is only one instance of TRACKER that communicates with UI, but in order to fulfill requirements from the client, there is a need for other modules to help. Based on what roles they play, I create 5 sub-clusters, which are entity, enum, error, operation and utility.

For classes in the cluster entity, they represent objects we operate on. They are phases and containers. Both have their own attributes, commands and queries.

For classes in the cluster enum, they represent constants. Since Eiffel doesn't have enum feature, I use singlrton design pattern to implement.

For the error cluster, it uses singleton design pattern to enumerate all possible errors in the project, all of which have different error code and message.

The cluster operation is for implementing undo/redo feature

Utility just provides useful and reusable utility classes. In this project, phase list and container list should be sorted, so QUIKSORT is very useful.

3. Table of modules — responsibilities and information hiding

In the cluster entity

1	PHASE	**Responsibility**: a representation of a phase
	Concrete	**Secret**: none

2	RADIATION_CONTAINER	**Responsibility**: a representation of a container
	Concrete	**Secret**: none

In the cluster enum

1	MATERIAL	**Responsibility**: a list of expected materials each of which has a unique integer code
	Concrete	**Secret**: none

In the cluster error

1	TRACKER_ERROR_ACCESS	**Responsibility**: containing all possible errors each of which is an instance of TRACKER_ERROR
	Concrete(expanded)	**Secret**: none

1.1	TRACKER_ERROR	**Responsibility**: a representation of an error which has a code and corresponding message
	Concrete	**Secret**: none

In the cluster operation

1	OPERATION	**Responsibility**: a representation of an operation that can be performed on TRACKER
	Abstract	**Secret**: some helper routines only for all descendants to use, like is_valid_string(str:STRING):BOOLEAN

1.1	NEW_TRACKER	**Responsibility**: an operation that set the tracker's two global constants: max_phase_radiation and max_container_radiation
	Concrete	**Secret**: none

1.2	NEW_PHASE	**Responsibility**: an operation of creating a new phase
	Concrete	**Secret**: none

1.3	NEW_CONTAINER	**Responsibility**: an operation of creating a new container
	Concrete	**Secret**: none

1.4	REMOVE_PHASE	**Responsibility**: an operation of removing an existing phase

		from the tracker
	Concrete	**Secret**: none

1.5	REMOVE_CONTAINER	**Responsibility**: an operation of removing a container from a specified phase
	Concrete	**Secret**: none

1.6	MOVE_CONTAINER	**Responsibility**: an operation of moving a container from one phase to another
	Concrete	**Secret**: none

In the cluster utility

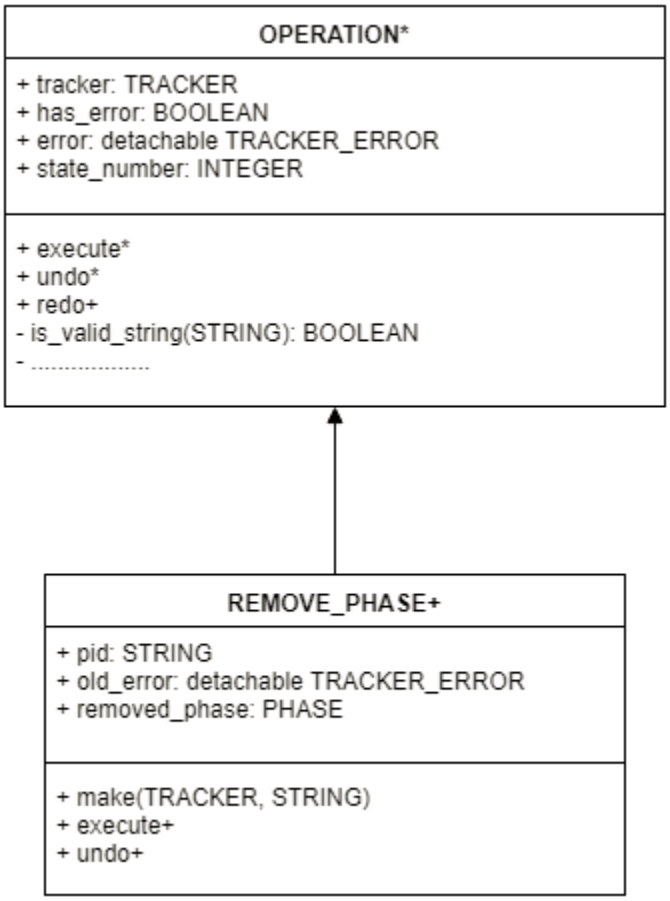
1	QUICKSORT[G -> COMPARABLE]	**Responsibility**: Sort the passed array
	Concrete	**Secret**: implemented with quick-sort algorithm

In the model cluster

1	TRACKER_ACCESS	**Responsibility**: providing one single instance of TRACKER to the user interface
	Concrete(expanded)	**Secret**: none

1.1	TRACKER	**Responsibility**: a representation of a tracker
	Concrete	**Secret**: implemented with a history list storing operations performed

4. Expanded description of design decisions



Based on the design of UI and undo/redo design pattern, I determine attributes all operations should store and behavior all should be able to perform on the abstract class-OPERATION. Since redo is straightforward in this project that just executes the operation again, I implement redo in the abstract class and it uses template design pattern. To implement undo, each operation should remember the old state of the tracker before execution, but considering space and time efficiency, I choose just to remember key part of the state, instead of the entire information, like above, REMOVE_PHASE just remembers old_error and removed_phase.

5. Significant Contracts (Correctness)

In the cluster entity

PHASE
Invariant: • pid_is_valid: identifiers/names must start with A-Z, a-z or 0..9 • phase_name_is_valid: identifiers/names must start with A-Z, a-z or 0..9 • expected_materials_are_valid: all_expected_materials are from MATERIAL • num_of_containers_within_capacity: # of containers is less than or equal to the phase capacity • materials_in_all_containers_are_expected: materials in all containers in this phase are expected

RADIATION_CONTAINER
Invariant: • cid_is_valid: identifiers/names must start with A-Z, a-z or 0..9 • material_is_valid: material contained is from MATERIAL • radioactivity_valid: radioactivity should be non-negative • in_the_phase: for data integrity, ensure the phase “pid” refers to has the container • material_is_expected_by_pid: material is expected by the phase “pid” refers to

6. Summary of Testing Procedures

Test file	Description	Passed
at1.txt	Normal scenario where no errors occur. set tracker radiation parameters, create some phases, add some containers, move some containers	✓
at2.txt	Test e11 and e10	✓
at3.txt	Test e1, e2, e3 and e4	✓
at4.txt	Test e5, e6, e7 and e8	✓
at5.txt	Test e11, e12, e14 and e15	✓

PASSED (35 out of 35)		
Case Type	**Passed**	**Total**
Violation	2	2
Boolean	33	33
All Cases	35	35
State	**Contract Violation**	**Test Name**
Test1	TRACKER_ERROR_TESTS	
PASSED	NONE	Test singleton pattern for tracker errors
Test2	MATERIAL_TESTS	
PASSED	NONE	Test material
Test3	PHASE_TESTS	
PASSED	NONE	test add_new_container, remove_container and radiation
PASSED	NONE	*test violation of capacity limit
PASSED	NONE	*test violation of material limit
Test4	QUICKSORT_TESTS	
PASSED	NONE	test QUICKSORT integers
PASSED	NONE	test QUICKSORT containers
Test5	NEW_TRACKER_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test failed execution, undo and redo
Test6	NEW_PHASE_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test e6: phase identifier already exists
PASSED	NONE	test e5: identifiers/names must start with A-Z, a-z or 0..9
PASSED	NONE	test e7: phase capacity must be a positive integer
PASSED	NONE	test e8: there must be at least one expected material for this phase

Test7	NEW_CONTAINER_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test e5: identifiers/names must start with A-Z, a-z or 0..9
PASSED	NONE	test e10: this container identifier already in tracker
PASSED	NONE	test e18: this container radiation must not be negative
PASSED	NONE	test e11: this container will exceed phase capacity
PASSED	NONE	test e14: container radiation capacity exceeded
PASSED	NONE	test e12: this container will exceed phase safe radiation
PASSED	NONE	test e13: phase does not expect this container material
Test8	REMOVE_PHASE_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test e1: current tracker is in use
PASSED	NONE	test e9: phase identifier not in the system
Test9	REMOVE_CONTAINER_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test e15: this container identifier not in tracker
Test10	MOVE_CONTAINER_TESTS	
PASSED	NONE	test successful execution, undo and redo
PASSED	NONE	test e15: this container identifier not in tracker
PASSED	NONE	teste 16: source and target phase identifier must be different
PASSED	NONE	teste e9: phase identifier not in the system
PASSED	NONE	teste e17: this container identifier is not in the source phase
PASSED	NONE	teste e11: this container will exceed phase capacity
PASSED	NONE	teste e12: this container will exceed phase safe radiation
PASSED	NONE	teste e13: phase does not expect this container material

7. Appendix (Contract view of all classes)

```
                -- Automatic generation produced by ISE Eiffel --
note
    description: "A tracker model"
    author: "Tao Wang"
    date: "$Date$"
    revision: "$Revision$"

class interface
    TRACKER

create {TRACKER_ACCESS, ES_TEST}
    make

feature -- model attributes

    state_num: INTEGER_32

    max_phase_rediation: VALUE

    max_container_radiation: VALUE

    error: detachable TRACKER_ERROR

    phases: ARRAY [PHASE]

    history: TWO_WAY_LIST [OPERATION]
                -- flags

    no_more_to_undo: BOOLEAN

    no_more_to_redo: BOOLEAN

    after_undo_or_redo: BOOLEAN

feature -- model operations

    state_num_update
                -- Perform update to the model state.

    reset
                -- Reset model state.

    new_tracker (m_p_r: VALUE; m_c_r: VALUE)

    new_phase (pid: STRING_8; phase_name: STRING_8; capacity: INTEGER_32;
expected_material: ARRAY [INTEGER_32])

    remove_phase (pid: STRING_8)

    new_container (cid: STRING_8; material_id: INTEGER_32; radiation: VALUE; pid: STRING_8)

    remove_container (cid: STRING_8)

    move_container (cid: STRING_8; pid1: STRING_8; pid2: STRING_8)

    undo

    redo

feature -- queries

    out: STRING_8
                -- New string containing terse printable representation
                -- of current object

    in_use: BOOLEAN
                -- Whether this tracker is in use
```

```
        has_phase (pid: STRING_8): BOOLEAN

        has_container (cid: STRING_8): BOOLEAN

        query_by_phase_id (pid: STRING_8): PHASE
                require
                        pid_exists: has_phase (pid)
                ensure
                        result_not_void: Result /= Void
                        result_correct: Result.pid ~ pid

        query_by_container_id (cid: STRING_8): RADIATION_CONTAINER
                require
                        cid_exists: has_container (cid)
                ensure
                        result_not_void: Result /= Void
                        result_correct: Result.cid ~ cid

        query_phase_by_container_id (cid: STRING_8): PHASE
                require
                        cid_exists: has_container (cid)
                ensure
                        result_not_void: Result /= Void
                        result_correct: across
                                        Result.containers as it
                                some
                                        it.item.cid ~ cid
                                end

    end -- class TRACKER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "An error that may occur when operating on a tracker."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        TRACKER_ERROR

create {TRACKER_ERROR_ACCESS}
        make

feature --Atributes

        error_code: INTEGER_32

        error_message: STRING_8

feature --queries

        out: STRING_8
                        -- New string containing terse printable representation
                        -- of current object

end -- class TRACKER_ERROR
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {TRACKER_ERROR_ACCESS}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

expanded class interface
        TRACKER_ERROR_ACCESS

create
        default_create

feature -- All errors

        E1: TRACKER_ERROR

        E2: TRACKER_ERROR

        E3: TRACKER_ERROR

        E4: TRACKER_ERROR

        E5: TRACKER_ERROR

        E6: TRACKER_ERROR

        E7: TRACKER_ERROR

        E8: TRACKER_ERROR

        E9: TRACKER_ERROR

        E10: TRACKER_ERROR

        E11: TRACKER_ERROR

        E12: TRACKER_ERROR

        E13: TRACKER_ERROR

        E14: TRACKER_ERROR

        E15: TRACKER_ERROR

        E16: TRACKER_ERROR

        E17: TRACKER_ERROR

        E18: TRACKER_ERROR

        E19: TRACKER_ERROR

        E20: TRACKER_ERROR

invariant
                E1 = E1
                E2 = E2
                E3 = E3
                E4 = E4
                E5 = E5
                E6 = E6
                E7 = E7
                E8 = E8
                E9 = E9
                E10 = E10
                E11 = E11
                E12 = E12
                E13 = E13
                E14 = E14
```

```
                E15 = E15
                E16 = E16
                E17 = E17
                E18 = E18

end -- class TRACKER_ERROR_ACCESS
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {REMOVE_PHASE}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        REMOVE_PHASE

create
        make

feature -- Attributes

        pid: STRING_8

        old_error: detachable TRACKER_ERROR

        removed_phase: detachable PHASE

feature -- Initialization

        make (t: TRACKER; p_id: STRING_8)

        execute

        undo
                ensure then
                        state_is_restored: tracker.error = old_error and ((not has_error)
implies across
                                                tracker.phases as it
                                        some
                                                it.item = removed_phase
                                        end)

end -- class REMOVE_PHASE
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {REMOVE_CONTAINER}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        REMOVE_CONTAINER

create
        make
```

```
feature -- Attributes

        cid: STRING_8

        old_error: detachable TRACKER_ERROR

        removed_container: detachable RADIATION_CONTAINER

        phase: detachable PHASE

feature -- Initialization

        make (t: TRACKER; c_id: STRING_8)

        execute

        undo

end -- class REMOVE_CONTAINER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "A container containing radioactive substance"
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        RADIATION_CONTAINER

create
        make

feature

        cid: STRING_8

        material_id: INTEGER_32

        radioactivity: VALUE

feature

        is_less alias "<" (other: like Current): BOOLEAN
                        -- implement the deferred feature in COMPARABLE

invariant
        material_id_is_valid: material_id > 0 and (create {MATERIAL}).List.count >= material_id
        cid_is_valid: cid.count /= 0 and then (('a' <= cid.at (1) and cid.at (1) <= 'z') or
('A' <= cid.at (1) and cid.at (1) <= 'Z') or ('0' <= cid.at (1) and cid.at (1) <= '9'))
        radioaivity_is_valid: radioactivity.as_double >= 0.00

end -- class RADIATION_CONTAINER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "A processing phase"
        author: ""
```

```
        date: "$Date$"
        revision: "$Revision$"

class interface
        PHASE

create
        make

feature -- attributes

        pid: STRING_8

        phase_name: STRING_8

        capacity: INTEGER_32

        expected_materials: ARRAY [INTEGER_32]

        containers: SEQ [RADIATION_CONTAINER]
                        --containers in the phase

feature -- commands

        add_new_container (container: RADIATION_CONTAINER)
                require
                        phase_is_not_full: containers.count < capacity
                        material_is_expected: expected_materials.has (container.material_id)
                ensure
                        container_is_added: containers.count = (old containers.count) + 1 and
containers.has (container)
                        containers_sorted: across
                                        1 |..| (containers.count - 1) as it
                                all
                                        containers [it.item].cid <= containers [it.item + 1].cid
                                end

        remove_container (container: RADIATION_CONTAINER)
                require
                        container_in_the_phase: containers.has (container)
                ensure
                        container_has_been_removed: containers.count = (old containers.count) -
1 and not containers.has (container)
                        containers_sorted: across
                                        1 |..| (containers.count - 1) as it
                                all
                                        containers [it.item].cid <= containers [it.item + 1].cid
                                end

feature -- queries

        radiation: VALUE
                        -- return the total amount of radioactivity of all containers

        is_less alias "<" (other: like Current): BOOLEAN
                        -- implement the deferred feature in COMPARABLE

        has_container (cid: STRING_8): BOOLEAN
                ensure
                                Result = across
                                        containers as it
                                some
                                        it.item.cid ~ cid
                                end

        is_full: BOOLEAN

invariant
        pid_is_valid: is_valid_string (pid)
        phase_name_is_valid: is_valid_string (phase_name)
```

```
        expected_materials_are_valid: across
                        expected_materials as c
                all
                        c.item >= 1 and c.item <= (create {MATERIAL}).List.count
                end
        num_of_containers_within_capacity: containers.count <= capacity
        materials_in_all_containers_are_expected: across
                        containers as c
                all
                        expected_materials.has (c.item.material_id)
                end

end -- class PHASE
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {OPERATION}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

deferred class interface
        OPERATION

feature -- attributes

        tracker: TRACKER

        has_error: BOOLEAN

        error: detachable TRACKER_ERROR

        state_number: INTEGER_32

feature -- commands

        execute

        undo

        redo

end -- class OPERATION
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {NEW_TRACKER}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        NEW_TRACKER

create
        make

feature -- Attributes
```

```
        max_phase_radiation: VALUE

        old_max_phase_radiation: VALUE

        max_container_radiation: VALUE

        old_max_container_radiation: VALUE

        old_error: detachable TRACKER_ERROR

feature -- Initialization

        make (t: TRACKER; mpr: VALUE; mcr: VALUE)

        execute

        undo

end -- class NEW_TRACKER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {NEW_PHASE}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        NEW_PHASE

create
        make

feature -- Attributes

        pid: STRING_8

        phase_name: STRING_8

        capacity: INTEGER_32

        expected_materials: ARRAY [INTEGER_32]

        old_error: detachable TRACKER_ERROR

        new_phase: detachable PHASE

feature -- Initialization

        make (t: TRACKER; id: STRING_8; p_n: STRING_8; c: INTEGER_32; e_m: ARRAY [INTEGER_32])

        execute

        undo
                ensure then
                        state_is_restored: tracker.error = old_error and across
                                        tracker.phases as it
                                all
                                        it.item /= new_phase
                                end

end -- class NEW_PHASE
                        -- Generated by ISE Eiffel --
```

```
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {NEW_CONTAINER}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        NEW_CONTAINER

create
        make

feature -- Attributes

        cid: STRING_8

        material_id: INTEGER_32

        radioactivity: VALUE

        pid: STRING_8

        old_error: detachable TRACKER_ERROR

        new_container: detachable RADIATION_CONTAINER

feature -- Initialization

        make (t: TRACKER; c_id: STRING_8; m_id: INTEGER_32; r: VALUE; p_id: STRING_8)

        execute

        undo
                ensure then
                        state_is_restored: tracker.error = old_error and ((not has_error)
implies across
                                        tracker.phases as it
                                all
                                        not it.item.has_container (cid)
                                end)

end -- class NEW_CONTAINER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {MOVE_CONTAINER}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

class interface
        MOVE_CONTAINER

create
        make

feature -- Attributes
```

```
        cid: STRING_8

        pid1: STRING_8

        pid2: STRING_8

        old_error: detachable TRACKER_ERROR

        moved_container: detachable RADIATION_CONTAINER

feature -- Initialization

        make (t: TRACKER; c_id: STRING_8; p_id1: STRING_8; p_id2: STRING_8)

        execute

        undo

end -- class MOVE_CONTAINER
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```

```
                        -- Automatic generation produced by ISE Eiffel --
note
        description: "Summary description for {MATERIAL}."
        author: ""
        date: "$Date$"
        revision: "$Revision$"

expanded class interface
        MATERIAL

create
        default_create

feature

        List: ARRAY [STRING_8]

feature --queries

        material_list_string (indices: ARRAY [INTEGER_32]): STRING_8
                require
                                indices.count /= 0
                                across
                                        indices as it
                                all
                                        it.item >= 1 and it.item <= List.count
                                end

invariant
                List = List

end -- class MATERIAL
                        -- Generated by ISE Eiffel --
                        -- For more details: http://www.eiffel.com --
```