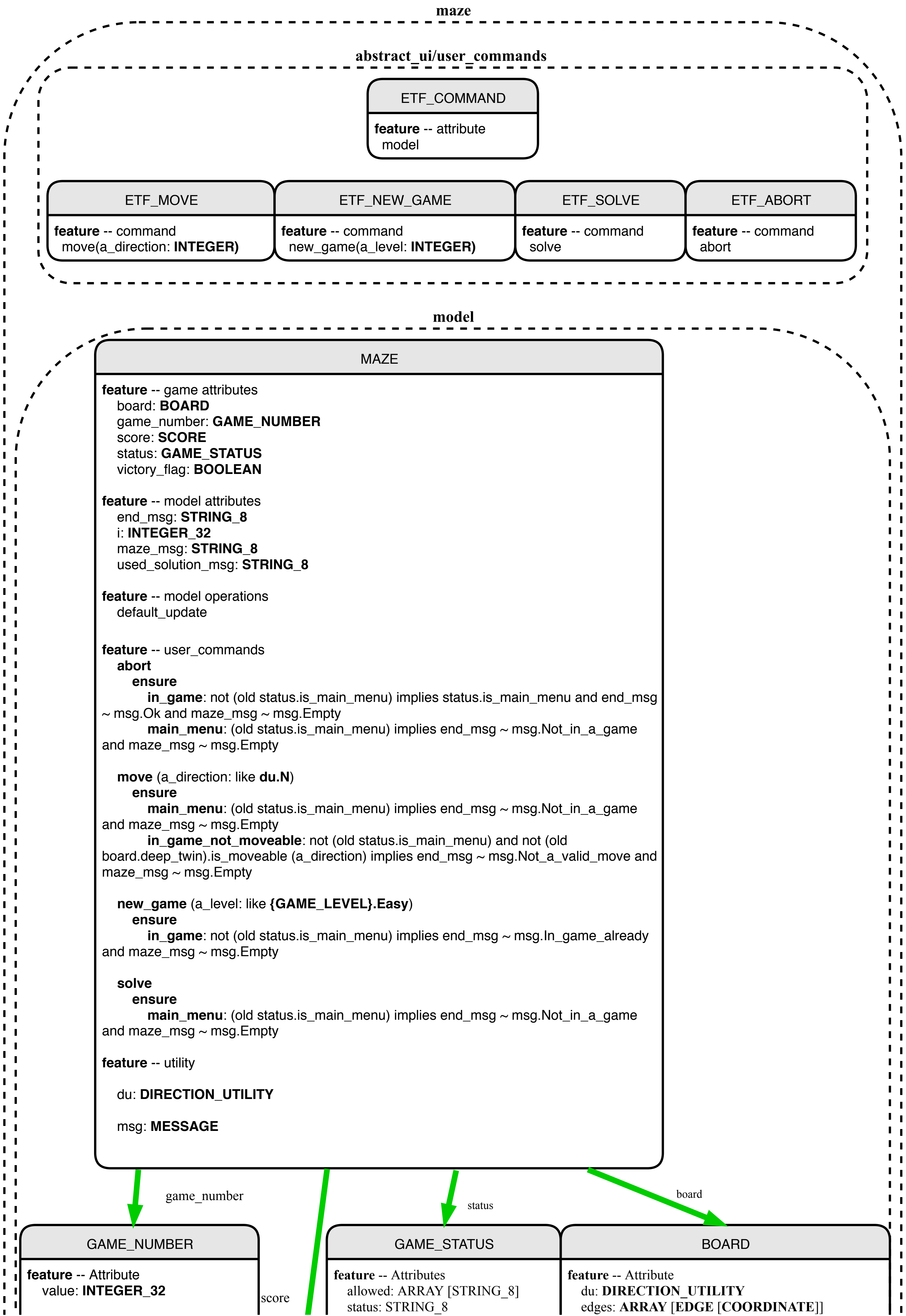


ETF_COMMAND

feature -- attribute
model

ETF_MOVE

feature -- command
move(a_direction: **INTEGER)**

ETF_NEW_GAME

feature -- command
new_game(a_level: **INTEGER)**

ETF_SOLVE

feature -- command
solve

ETF_ABORT

feature -- command
abort

MAZE

feature -- game attributes
board: **BOARD**
game_number: **GAME_NUMBER**
score: **SCORE**
status: **GAME_STATUS**
victory_flag: **BOOLEAN**

feature -- model attributes
end_msg: **STRING_8**
i: **INTEGER_32**
maze_msg: **STRING_8**
used_solution_msg: **STRING_8**

feature -- model operations
default_update

feature -- user_commands
abort
ensure
in_game: not (old status.is_main_menu) implies status.is_main_menu and end_msg ~ msg.Ok and maze_msg ~ msg.Empty
main_menu: (old status.is_main_menu) implies end_msg ~ msg.Not_in_a_game and maze_msg ~ msg.Empty

move (a_direction: like **du.N**)
ensure
main_menu: (old status.is_main_menu) implies end_msg ~ msg.Not_in_a_game and maze_msg ~ msg.Empty
in_game_not_moveable: not (old status.is_main_menu) and not (old board.deep_twin).is_moveable (a_direction) implies end_msg ~ msg.Not_a_valid_move and maze_msg ~ msg.Empty

new_game (a_level: like **{GAME_LEVEL}.Easy**)
ensure
in_game: not (old status.is_main_menu) implies end_msg ~ msg.In_game_already and maze_msg ~ msg.Empty

solve
ensure
main_menu: (old status.is_main_menu) implies end_msg ~ msg.Not_in_a_game and maze_msg ~ msg.Empty

feature -- utility

du: **DIRECTION_UTILITY**

msg: **MESSAGE**

GAME_NUMBER

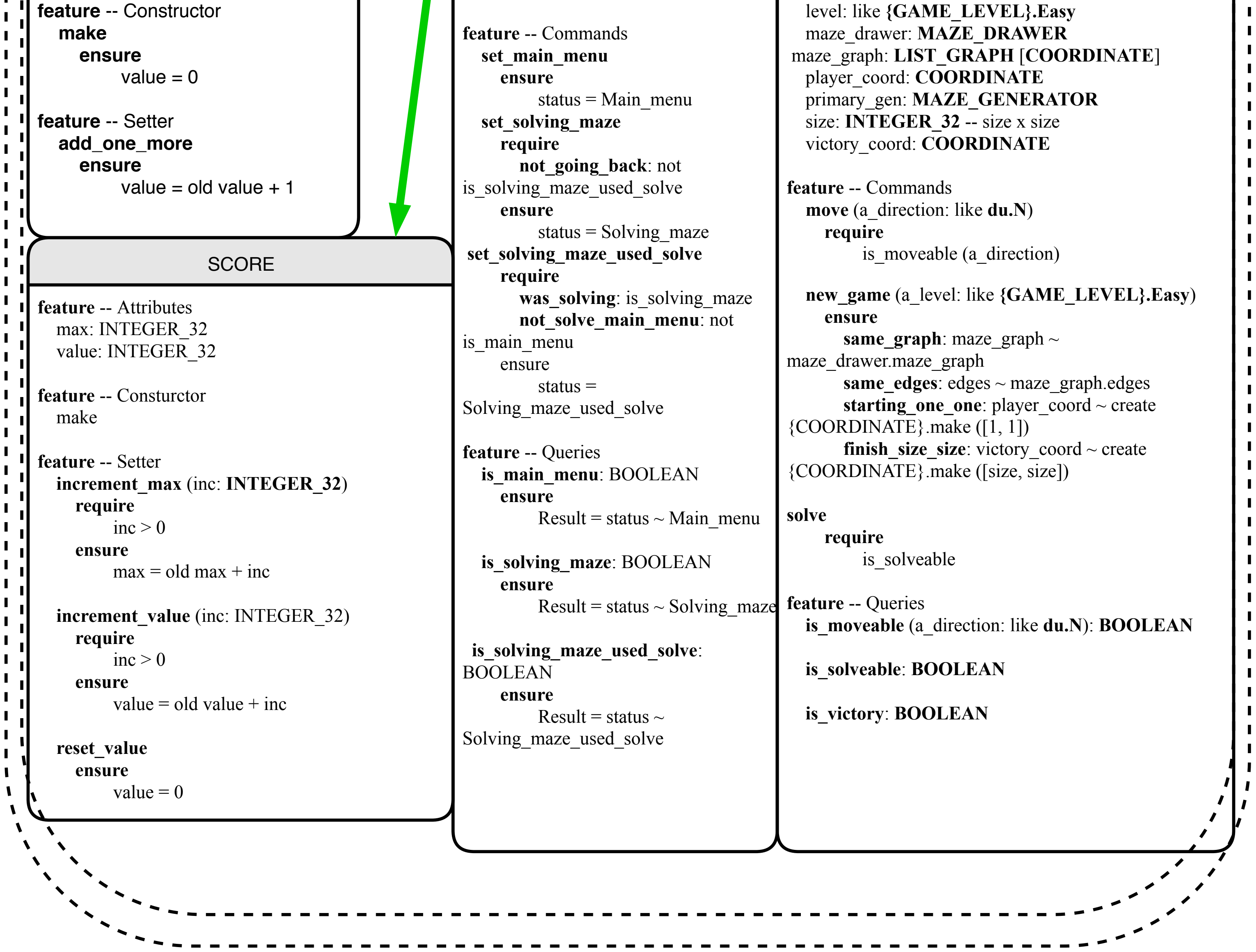
feature -- Attribute
value: **INTEGER_32**

GAME_STATUS

feature -- Attributes
allowed: ARRAY [STRING_8]
status: STRING_8

BOARD

feature -- Attribute
du: **DIRECTION_UTILITY**
edges: **ARRAY [EDGE [COORDINATE]]**



```
feature -- Constructor
  make
    ensure
        value = 0

feature -- Setter
  add_one_more
    ensure
        value = old value + 1
```

SCORE

```
feature -- Attributes
  max: INTEGER_32
  value: INTEGER_32

feature -- Consturctor
  make

feature -- Setter
  increment_max (inc: INTEGER_32)
    require
        inc > 0
    ensure
        max = old max + inc

  increment_value (inc: INTEGER_32)
    require
        inc > 0
    ensure
        value = old value + inc

  reset_value
    ensure
        value = 0
```

```
feature -- Commands
  set_main_menu
    ensure
        status = Main_menu
  set_solving_maze
    require
      not_going_back: not
is_solving_maze_used_solve
    ensure
        status = Solving_maze
 set_solving_maze_used_solve
    require
      was_solving: is_solving_maze
      not_solve_main_menu: not
is_main_menu
    ensure
        status =
Solving_maze_used_solve

feature -- Queries
  is_main_menu: BOOLEAN
    ensure
        Result = status ~ Main_menu

  is_solving_maze: BOOLEAN
    ensure
        Result = status ~ Solving_maze

 is_solving_maze_used_solve:
BOOLEAN
    ensure
        Result = status ~
Solving_maze_used_solve
```

```
  level: like {GAME_LEVEL}.Easy
  maze_drawer: MAZE_DRAWER
 maze_graph: LIST_GRAPH [COORDINATE]
  player_coord: COORDINATE
  primary_gen: MAZE_GENERATOR
  size: INTEGER_32 -- size x size
  victory_coord: COORDINATE

feature -- Commands
  move (a_direction: like du.N)
    require
        is_moveable (a_direction)

  new_game (a_level: like {GAME_LEVEL}.Easy)
    ensure
      same_graph: maze_graph ~
maze_drawer.maze_graph
      same_edges: edges ~ maze_graph.edges
      starting_one_one: player_coord ~ create
{COORDINATE}.make ([1, 1])
      finish_size_size: victory_coord ~ create
{COORDINATE}.make ([size, size])

solve
    require
        is_solveable

feature -- Queries
  is_moveable (a_direction: like du.N): BOOLEAN

  is_solveable: BOOLEAN

  is_victory: BOOLEAN
```