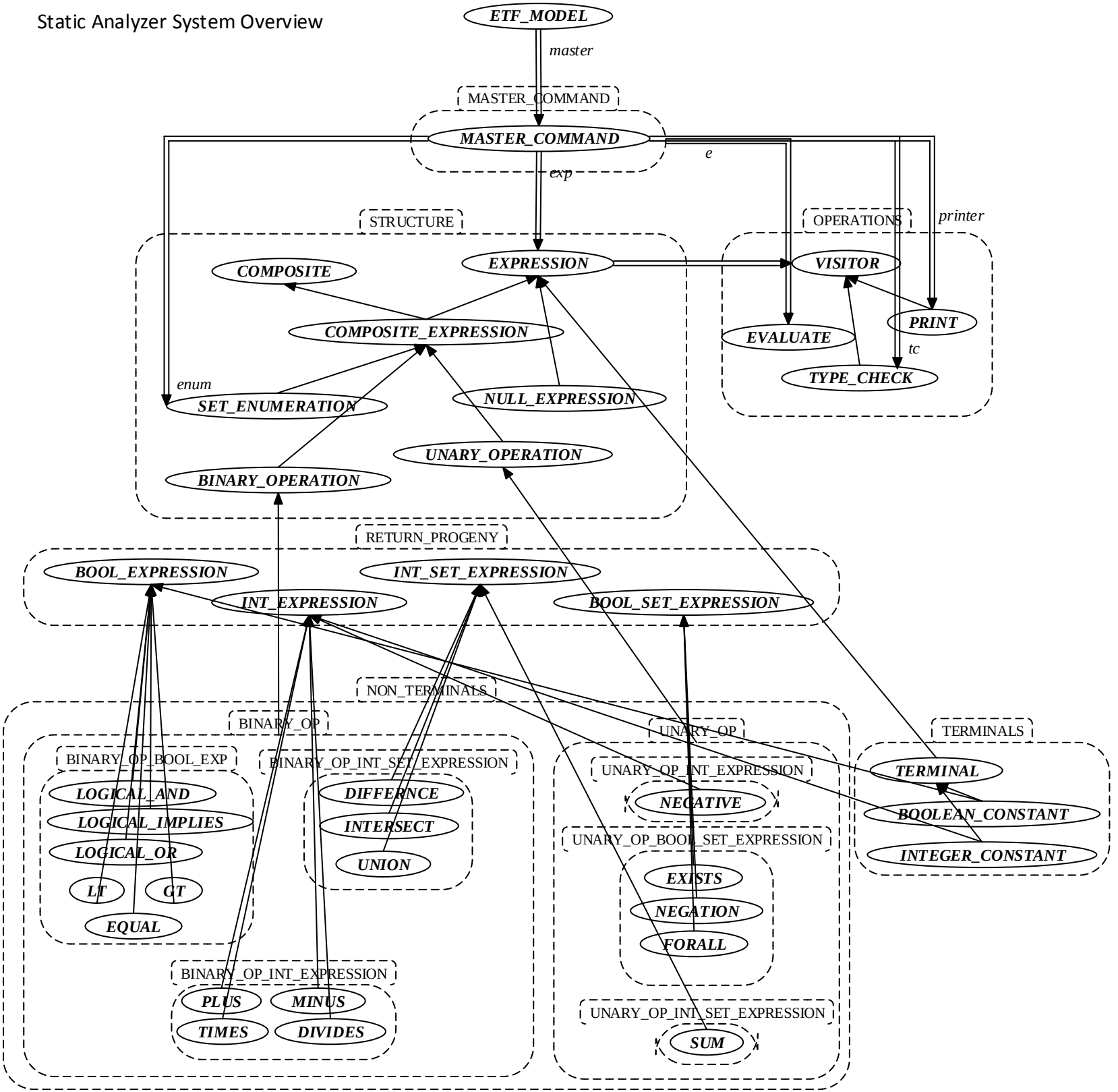
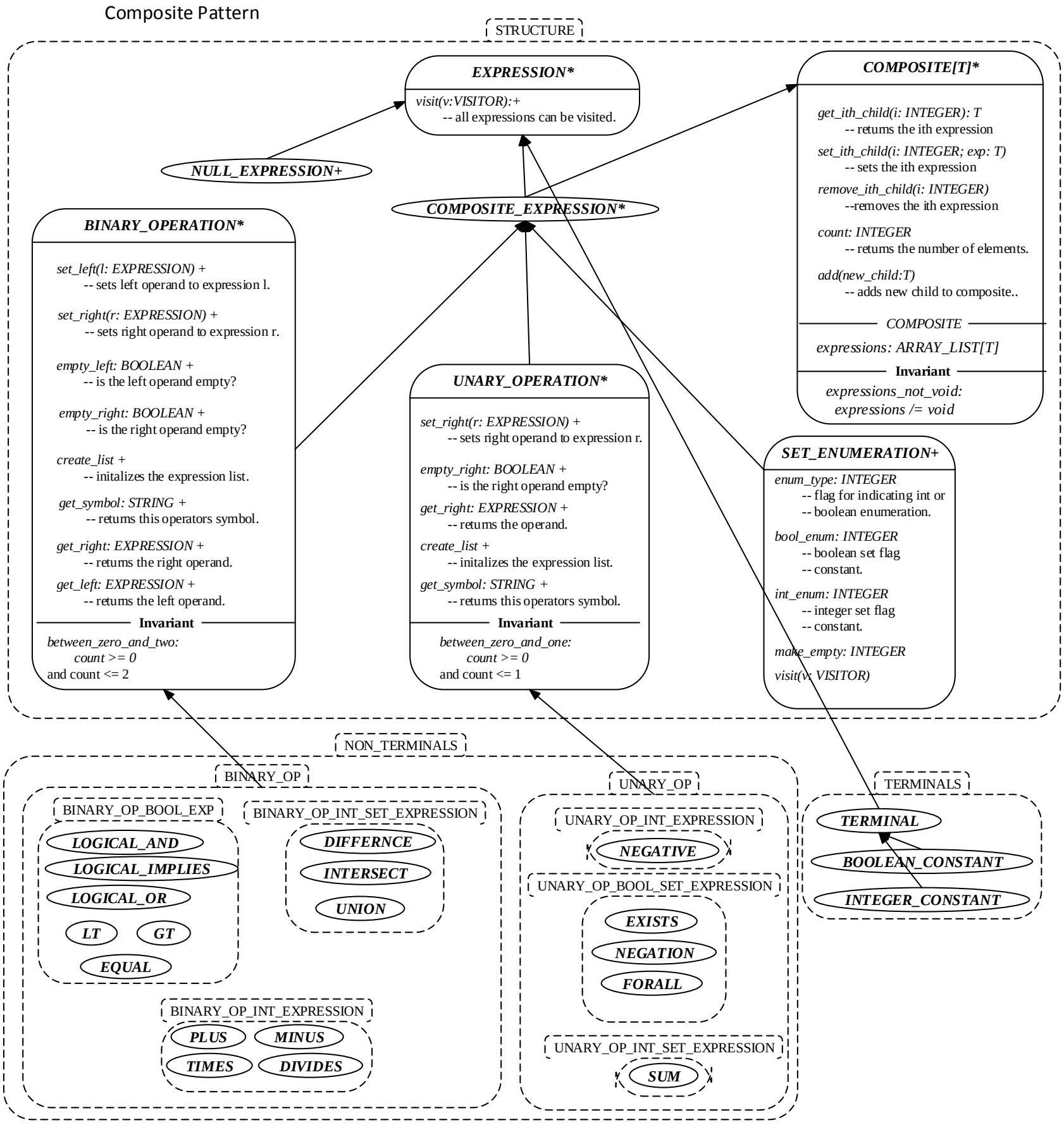
Course: EECS 3311 Software Design
Semester: Summer 2015
Name: Christopher Kerley
CSE login: cse23056
Submittion Login: cse23056

Static Analyzer System Overview




Composite Pattern
STRUCTURE
EXPRESSION*
visit(v:VISITOR):+
-- all expressions can be visited.
NULL_EXPRESSION+
COMPOSITE_EXPRESSION*
COMPOSITE[T]*
get_ith_child(i: INTEGER): T
-- returns the ith expression
set_ith_child(i: INTEGER; exp: T)
-- sets the ith expression
remove_ith_child(i: INTEGER)
--removes the ith expression
count: INTEGER
-- returns the number of elements.
add(new_child:T)
-- adds new child to composite..
COMPOSITE
expressions: ARRAY_LIST[T]
Invariant
expressions_not_void:
expressions /= void
BINARY_OPERATION*
set_left(l: EXPRESSION) +
-- sets left operand to expression l.
set_right(r: EXPRESSION) +
-- sets right operand to expression r.
empty_left: BOOLEAN +
-- is the left operand empty?
empty_right: BOOLEAN +
-- is the right operand empty?
create_list +
-- initalizes the expression list.
get_symbol: STRING +
-- returns this operators symbol.
get_right: EXPRESSION +
-- returns the right operand.
get_left: EXPRESSION +
-- returns the left operand.
Invariant
between_zero_and_two:
count >= 0
and count <= 2
UNARY_OPERATION*
set_right(r: EXPRESSION) +
-- sets right operand to expression r.
empty_right: BOOLEAN +
-- is the right operand empty?
get_right: EXPRESSION +
-- returns the operand.
create_list +
-- initalizes the expression list.
get_symbol: STRING +
-- returns this operators symbol.
Invariant
between_zero_and_one:
count >= 0
and count <= 1
SET_ENUMERATION+
enum_type: INTEGER
-- flag for indicating int or
-- boolean enumeration.
bool_enum: INTEGER
-- boolean set flag
-- constant.
int_enum: INTEGER
-- integer set flag
-- constant.
make_empty: INTEGER
visit(v: VISITOR)
NON_TERMINALS
BINARY_OP
BINARY_OP_BOOL_EXP
LOGICAL_AND
LOGICAL_IMPLIES
LOGICAL_OR
LT
GT
EQUAL
BINARY_OP_INT_SET_EXPRESSION
DIFFERNCE
INTERSECT
UNION
BINARY_OP_INT_EXPRESSION
PLUS
MINUS
TIMES
DIVIDES
UNARY_OP
UNARY_OP_INT_EXPRESSION
NEGATIVE
UNARY_OP_BOOL_SET_EXPRESSION
EXISTS
NEGATION
FORALL
UNARY_OP_INT_SET_EXPRESSION
SUM
TERMINALS
TERMINAL
BOOLEAN_CONSTANT
INTEGER_CONSTANT

Visitor Pattern

OPERATIONS

TYPE_CHECK+

visit_null_expression(n: NULL_EXPRESSION) +
visit_(n: DIFFERENCE) +
visit_(n: DIVIDES) +
visit_(n: EQUAL) +
visit_(n:GT) +
visit_(n: INTERSECT) +
visit_(n: LOGICAL_AND) +
visit_(n: LOGICAL_IMPLIES) +
visit_(n: LOGICAL_OR) +
visit_(n: LT) +
visit_(n:MINUS) +
visit_(n:PLUS) +
visit_(n: TIMES) +
visit_(n: UNION) +
visit_(n:EXISTS) +
visit_(n: FORALL) +
visit_(n: NEGATION) +
visit_(n: NEGATIVE) +
visit_(n: SUM) +
visit_(n: SET_ENUMERATION) +
visit_(n: BOOLEAN_CONSTANT) +
visit_(n: INTEGER_CONSTANT) +

value: BOOLEAN
-- holds if it is type correct.
make
-- type check constructor
reset
-- resets type checker.

NONE

unspecified: BOOLEAN

is_type_correct(e: EXPRESSION): BOOLEAN
-- recursively checks type.

PRINT+

visit_null_expression(n: NULL_EXPRESSION) +
visit_(n: DIFFERENCE) +
visit_(n: DIVIDES) +
visit_(n: EQUAL) +
visit_(n:GT) +
visit_(n: INTERSECT) +
visit_(n: LOGICAL_AND) +
visit_(n: LOGICAL_IMPLIES) +
visit_(n: LOGICAL_OR) +
visit_(n: LT) +
visit_(n:MINUS) +
visit_(n:PLUS) +
visit_(n: TIMES) +
visit_(n: UNION) +
visit_(n:EXISTS) +
visit_(n: FORALL) +
visit_(n: NEGATION) +
visit_(n: NEGATIVE) +
visit_(n: SUM) +
visit_(n: SET_ENUMERATION) +
visit_(n: BOOLEAN_CONSTANT) +
visit_(n: INTEGER_CONSTANT) +

value: STRING
-- printout string
make
-- print constructor
reset
-- set value to empty

NONE

unspecified: BOOLEAN

r_print(e: EXPRESSION): STRING
-- recursively builds print value.

VISITOR*

visit_null_expression(n: NULL_EXPRESSION) *
visit_(n: DIFFERENCE) *
visit_(n: DIVIDES) *
visit_(n: EQUAL) *
visit_(n:GT) *
visit_(n: INTERSECT) *
visit_(n: LOGICAL_AND) *
visit_(n: LOGICAL_IMPLIES) *
visit_(n: LOGICAL_OR) *
visit_(n: LT) *
visit_(n:MINUS) *
visit_(n:PLUS) *
visit_(n: TIMES) *
visit_(n: UNION) *
visit_(n:EXISTS) *
visit_(n: FORALL) *
visit_(n: NEGATION) *
visit_(n: NEGATIVE) *
visit_(n: SUM) *
visit_(n: SET_ENUMERATION) *
visit_(n: BOOLEAN_CONSTANT) *
visit_(n: INTEGER_CONSTANT) *

EVALUATE+

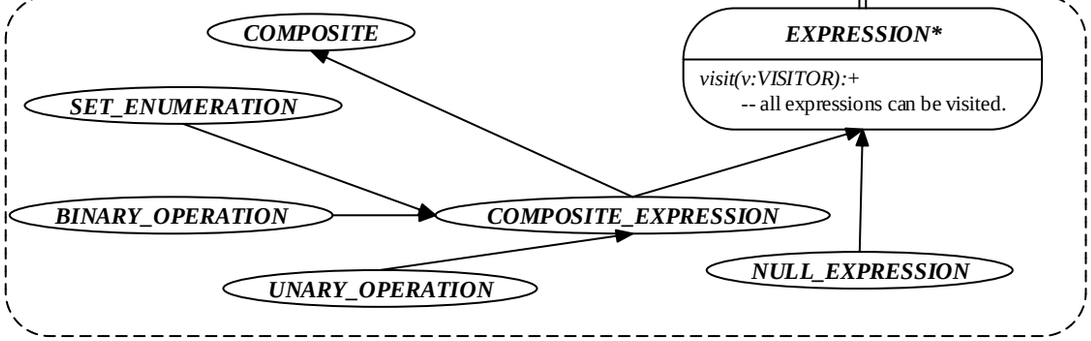
remove_duplicates(set: SET_ENUMERATION): SET_ENUMERATION

value: BOOLEAN
-- holds if it is type correct.
make_eval(e: EXPRESSION)
-- type check constructor

NONE

evaluate(e: EXPRESSION): EXPRESSION
-- recursively evaluates expression.

STRUCTURE



Note: Evaluate does not inherit from VISITOR because all types of expressions are evaulate by one recursive function. In other words EVALUATE Is a ultility class.

Concise views of the composite side or the pattern are used because expanded views are already shown on the previous page.