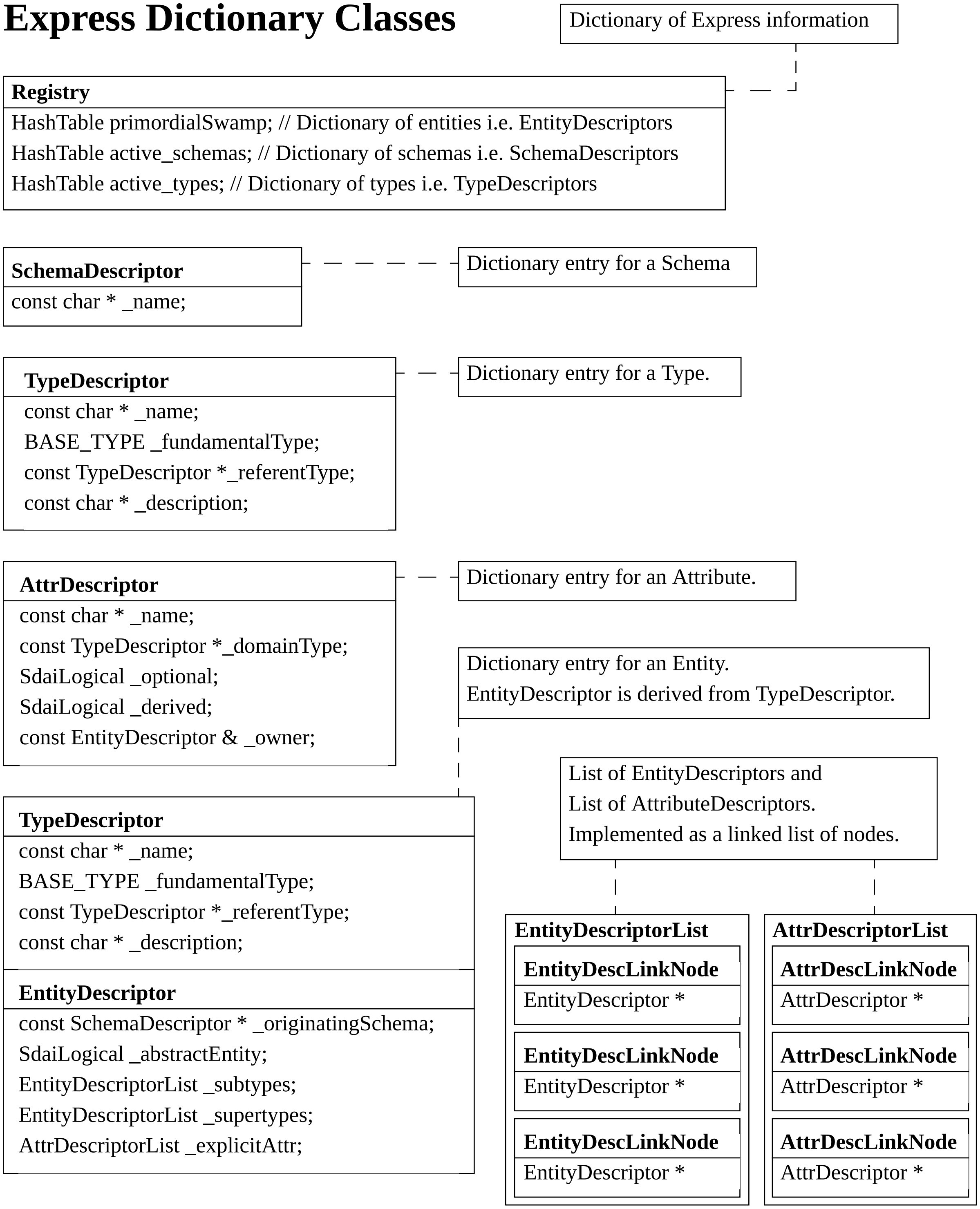
Express Dictionary Classes



Dictionary of Express information

Registry

HashTable primordialSwamp; // Dictionary of entities i.e. EntityDescriptors
HashTable active_schemas; // Dictionary of schemas i.e. SchemaDescriptors
HashTable active_types; // Dictionary of types i.e. TypeDescriptors

SchemaDescriptor

const char * _name;

Dictionary entry for a Schema

TypeDescriptor

const char * _name;
BASE_TYPE _fundamentalType;
const TypeDescriptor *_referentType;
const char * _description;

Dictionary entry for a Type.

AttrDescriptor

const char * _name;
const TypeDescriptor *_domainType;
SdaiLogical _optional;
SdaiLogical _derived;
const EntityDescriptor & _owner;

Dictionary entry for an Attribute.

Dictionary entry for an Entity.
EntityDescriptor is derived from TypeDescriptor.

TypeDescriptor

const char * _name;
BASE_TYPE _fundamentalType;
const TypeDescriptor *_referentType;
const char * _description;

EntityDescriptor

const SchemaDescriptor * _originatingSchema;
SdaiLogical _abstractEntity;
EntityDescriptorList _subtypes;
EntityDescriptorList _supertypes;
AttrDescriptorList _explicitAttr;

List of EntityDescriptors and
List of AttributeDescriptors.
Implemented as a linked list of nodes.

EntityDescriptorList

EntityDescLinkNode
EntityDescriptor *

EntityDescLinkNode
EntityDescriptor *

EntityDescLinkNode
EntityDescriptor *

AttrDescriptorList

AttrDescLinkNode
AttrDescriptor *

AttrDescLinkNode
AttrDescriptor *

AttrDescLinkNode
AttrDescriptor *

Classes Used for Entity Instances

```
STEPattribute
ErrorDescriptor _error;
unsigned int _derive;
const AttrDescriptor * AttrDescriptor;
union {
      SdaiInteger *i;
      class SdaiString *S;
      class SdaiBinary *b;
      SdaiReal *r;
      class STEPentity *c;
      class STEPaggregate *a;
      class STEPenumeration *e;
      class SdaiSelect *sh;
      class SCLundefined *u;
} ptr;
```

An element in STEPentity’s list of attributes.

_error - may contain an error level and message associated with the attribute value.

_derive - boolean specifying whether the attribute has had its value overridden by the instantiation of an inheriting entity which contains a DERIVE statement in the Express definition. If true the attribute is not allowed to have a value and Part 21 specifies that an asterisk represent its value in an Exchange File. This may be true or false depending on which child entity has been instantiated (the one containing the DERIVE attribute or not).

AttrDescriptor - points at the Express dictionary entry for this attribute. An Enum variable found in the dictionary entry specifies the attribute type and is used to know how to access and use the ptr member variable.

ptr - points at the data storage area defined by the generated entity class to represent the attribute’s data value.

```
STEPentity
STEPattributeList attributes;
int STEPfile_id;
ErrorDescriptor _error;
const EntityDescriptor * EntityDescriptor;
```

Base class of all generated entity classes.

attributes - is a list that provides a generic way to access and manipulate the attributes associated with the derived entity class.

STEPfile_id - is the numeric part of what Part 21 calls ENTITY_NAME.

_error - may contain an error level and message associated with the entity instance.

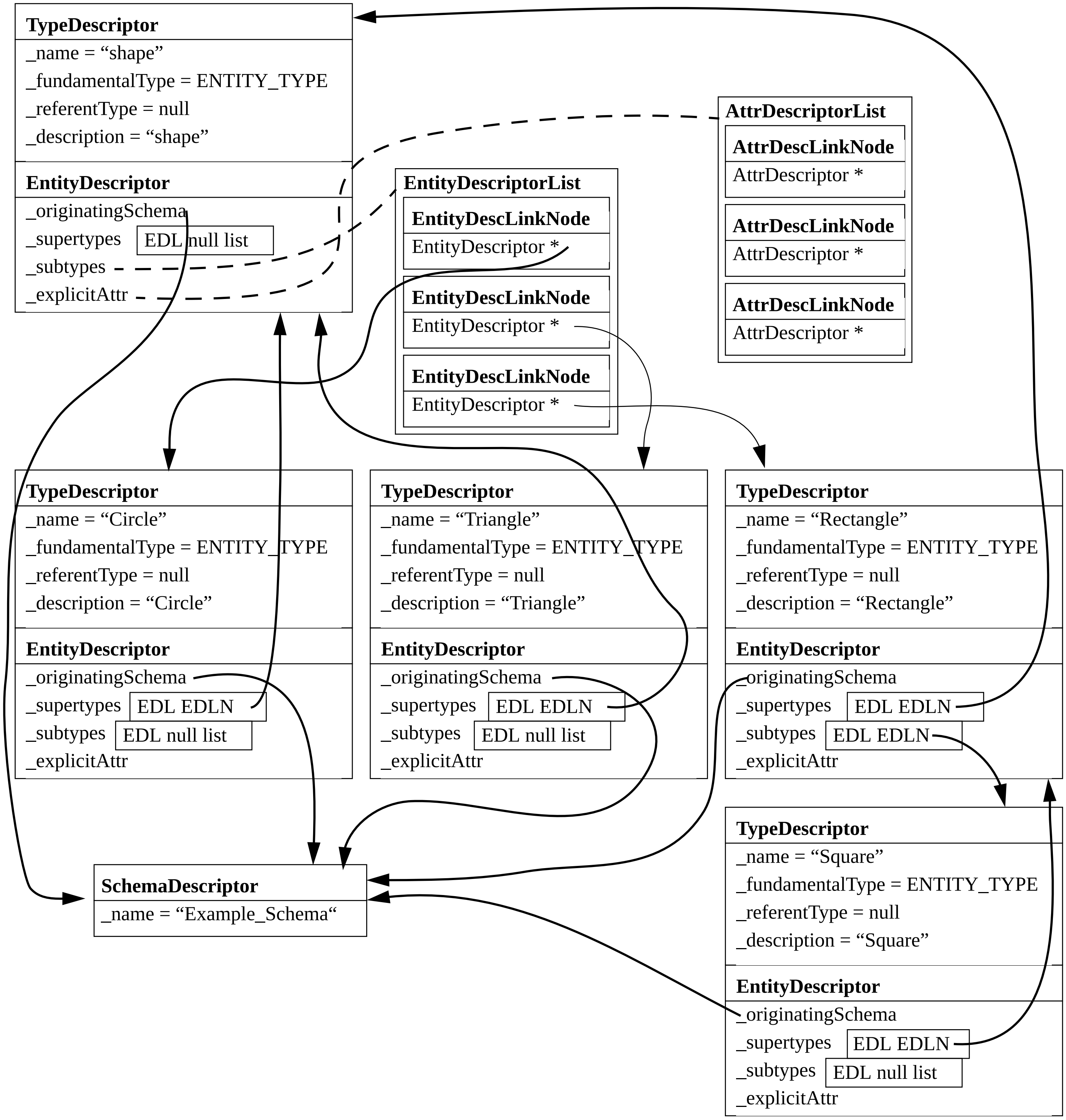
EntityDescriptor - is a pointer to the Express dictionary entry for the entity.

```
STEPattrList
  AttrListNode
  STEPattribute *attr;

  AttrListNode
  STEPattribute *attr;

  AttrListNode
  STEPattribute *attr;
```

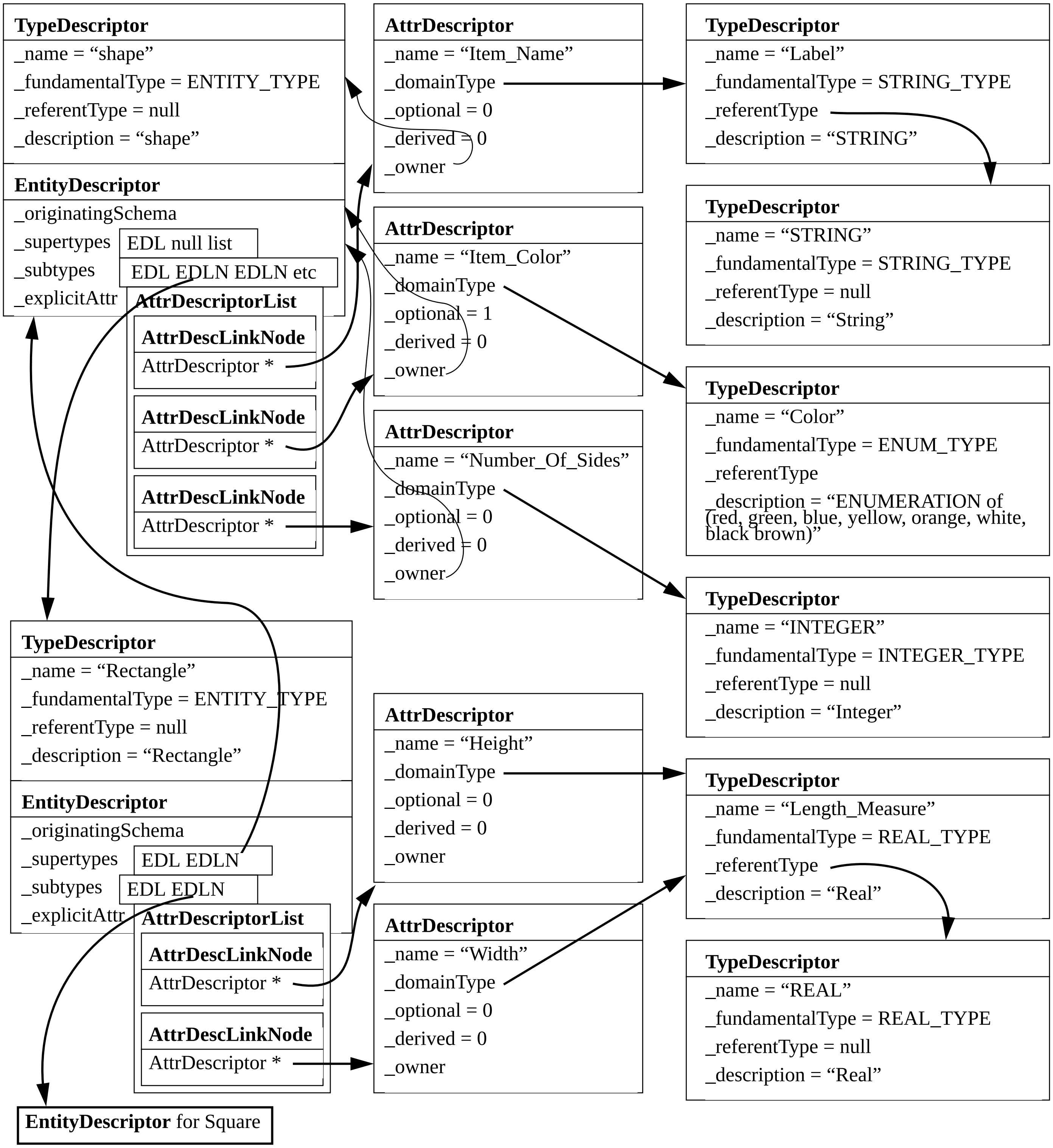
List of STEPattributes.

Implemented as a linked list of nodes.

Express Dictionary Classes

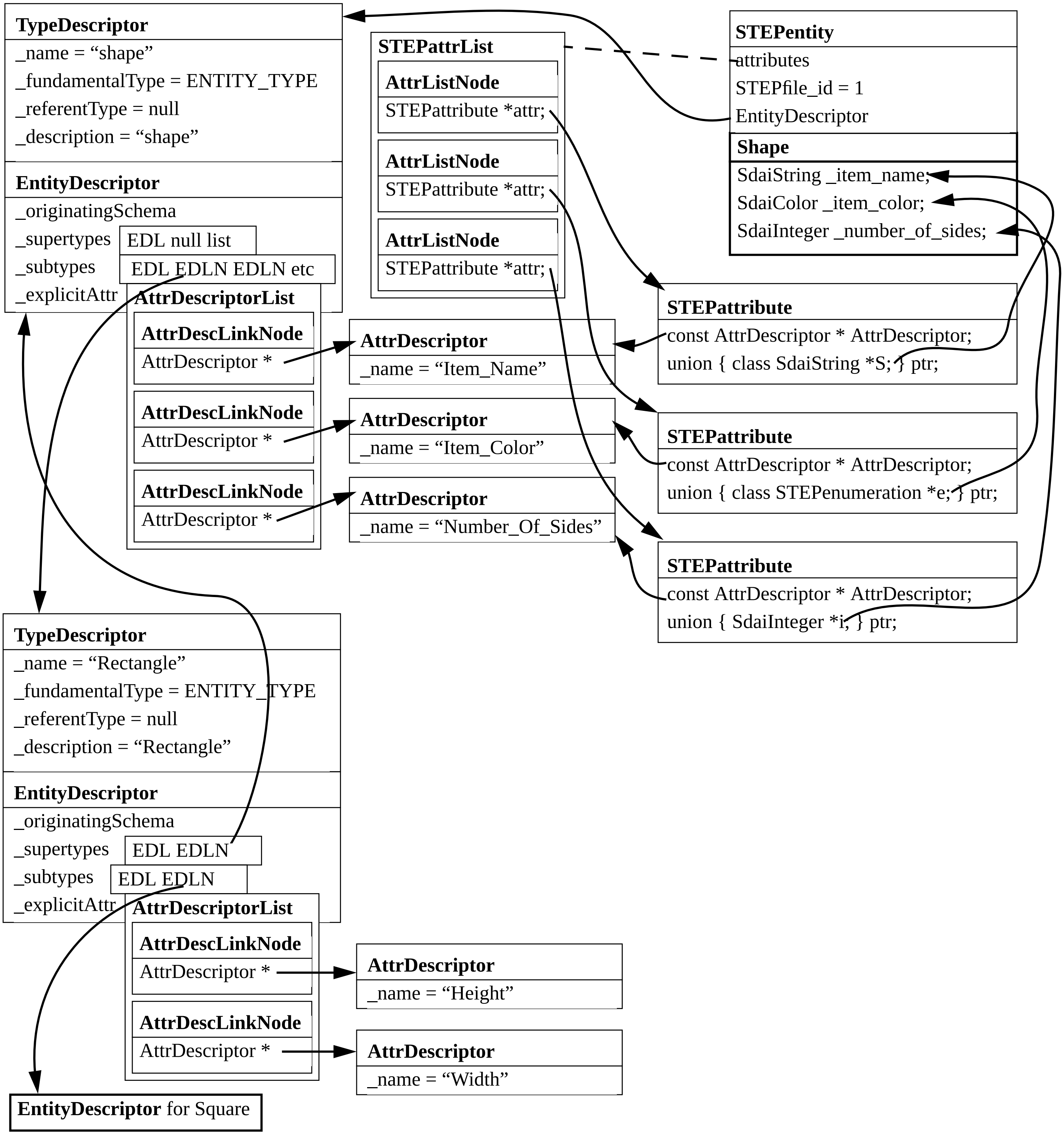


Express Dictionary Classes

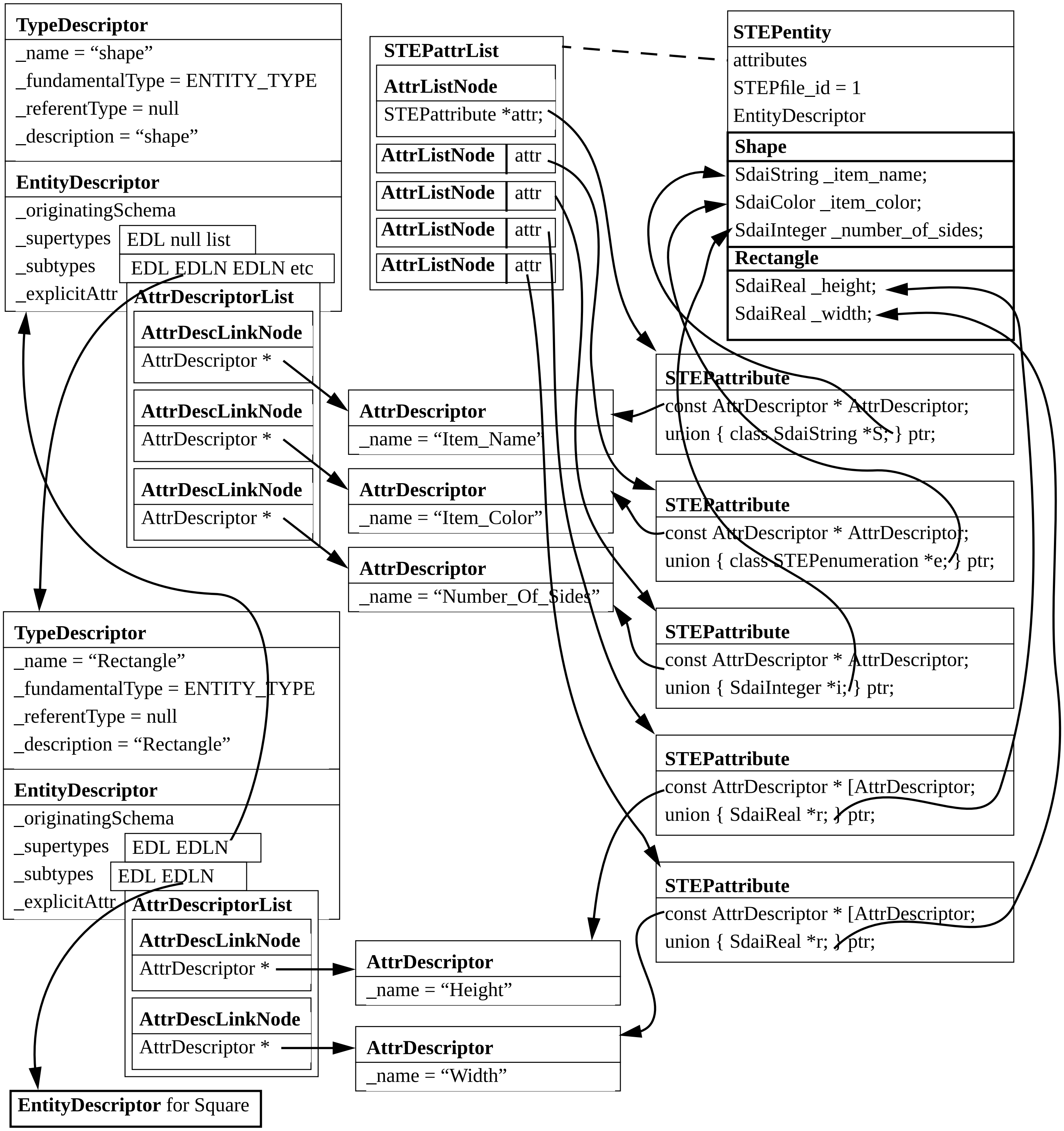


]

Express Dictionary and Entity Instance Classes



Express Dictionary and Entity Instance Classes



Classes for STEPattribute Instance Data Values

STEPentity

STEPaggregate, STEPnode

STEPenumeration

SdaiSelect

SdaiString

SdaiBinary

SCLundefined

Virtual Functions for the above classes

STEPread() - read Exchange File format

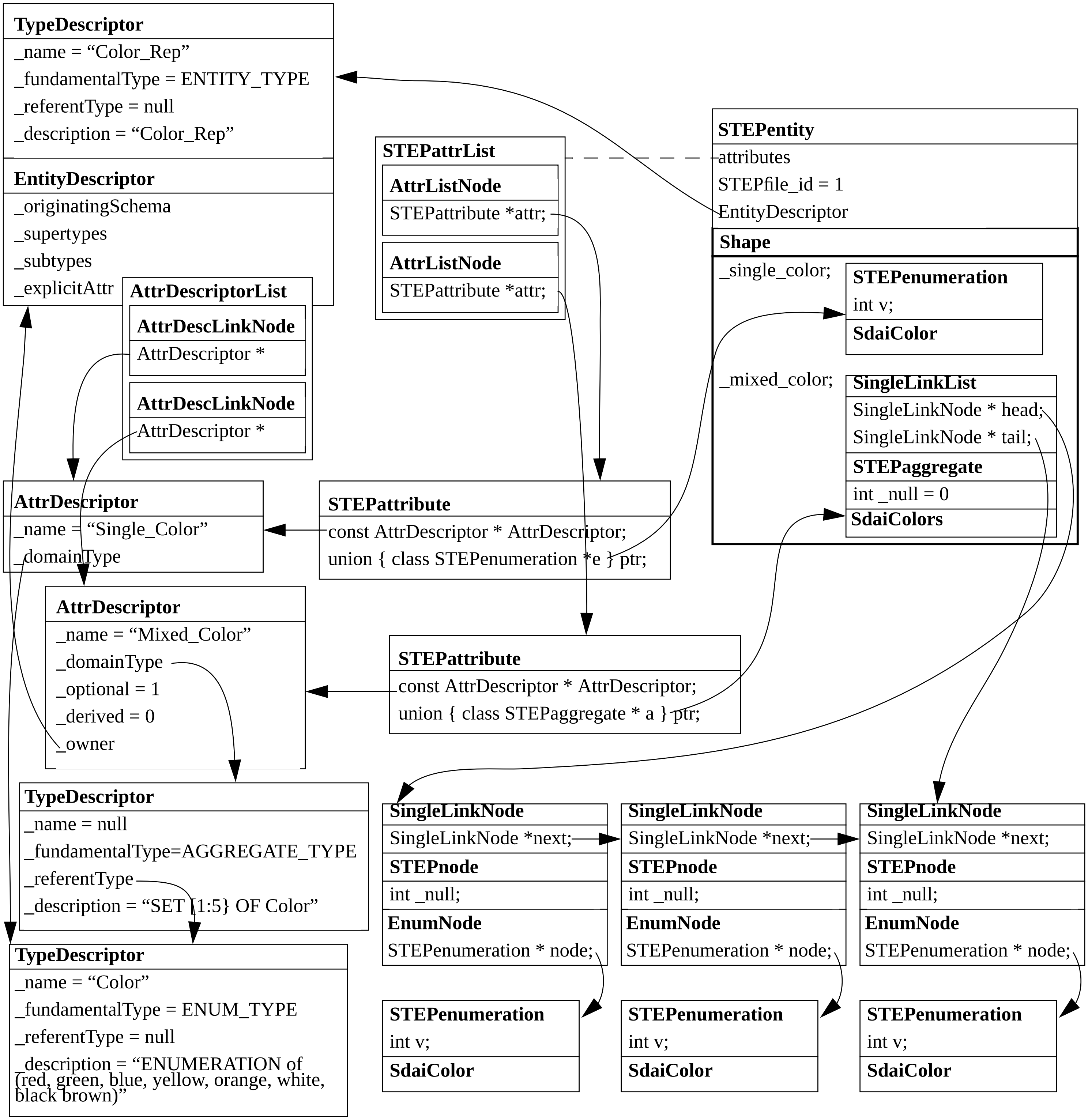
STEPwrite() - write Exchange File format

StrToVal() - read user interface format

asStr() - write user interface format

Validate functions

Express Dictionary and Entity Instance Classes



Express Used in These Notes

```
SCHEMA example_schema;

TYPE label = STRING;
END_TYPE;

TYPE color = ENUMERATION OF (red, green, blue, yellow, orange, white, black, brown);
END_TYPE;

TYPE length_measure = REAL;
END_TYPE;

TYPE point = REAL;
END_TYPE;

ENTITY color_rep;
    single_color : OPTIONAL color;
    mixed_color : OPTIONAL set [1:5] of color;
END_ENTITY;

ENTITY shape
SUPERTYPE OF (ONEOF (circle, triangle, rectangle));
    item_name : label;
    item_color : OPTIONAL color;
    number_of_sides : INTEGER;
END_ENTITY;

ENTITY circle
SUBTYPE OF (shape);
    radius : real;
END_ENTITY;

ENTITY triangle
SUBTYPE OF (shape);
    side1_length, side2_length, side3_length : length_measure;
END_ENTITY;

ENTITY rectangle
SUPERTYPE OF (square)
SUBTYPE OF (shape);
    height : length_measure;
    width : length_measure;
END_ENTITY;
```

Express Used in These Notes (continued)

```
ENTITY square
SUBTYPE OF (rectangle);
END_ENTITY;

ENTITY cartesian_point;
    x : point;
    y : point;
    z : OPTIONAL point;
END_ENTITY;

ENTITY line;
    end_point_one : cartesian_point;
    end_point_two : cartesian_point;
END_ENTITY;

ENTITY poly_line;
    points : LIST OF line;
END_ENTITY;

END_SCHEMA;
```