# Eclipse Modelling Framework : Basic OCL and Testing

Sagar Sen (2nd Year PhD student, INRIA, Rennes)

Date: 16th July, 2008

Venue: Modelling and Simulation Design Lab, McGill University

# Object Constraint Language

- *Side-effect free* constraint specification language
- Concrete Textual Syntax : ".ocl" file
- Specification of constraints in an ECore model.
- ECore + OCL = Declarative specification of Modelling Language

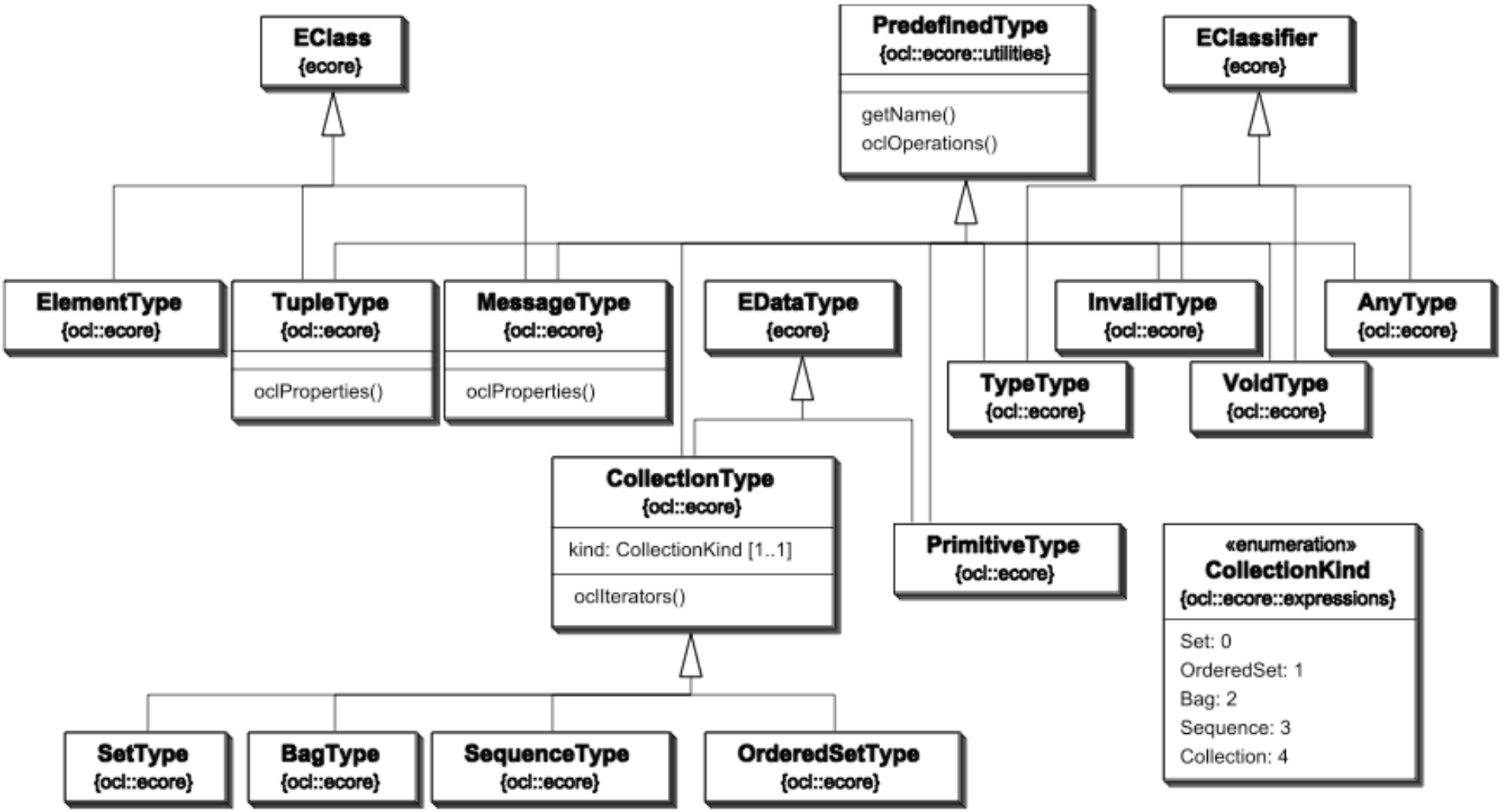
# What can Basic OCL (for ECore2Alloy) do?

- As a query language
- To specify invariants on classes and types in the class model
- To specify type invariant for Stereotypes
- To describe pre- and post conditions on Operations and Methods
- To describe Guards
- To specify target (sets) for messages and actions
- To specify constraints on operations
- To specify derivation rules for attributes for any expression over a UML model

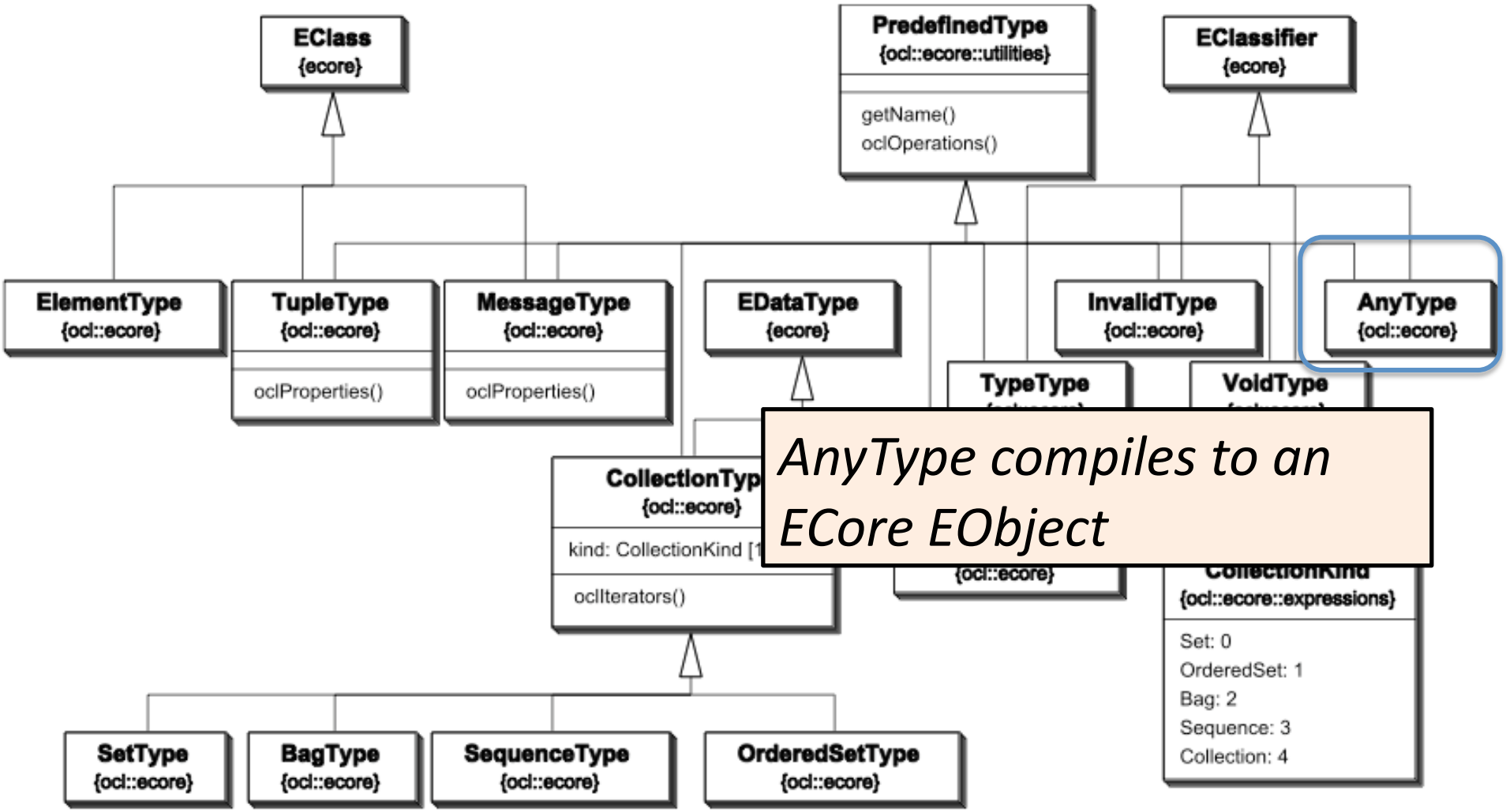
# Outline

- Basic/Essential OCL Types
- Basic/Essential OCL Expressions
- OCL Standard Library : Primitive Types
- OCL Standard Library : Collection Types
- OCL Standard Library: Operations
- *Test OCL Expressions*

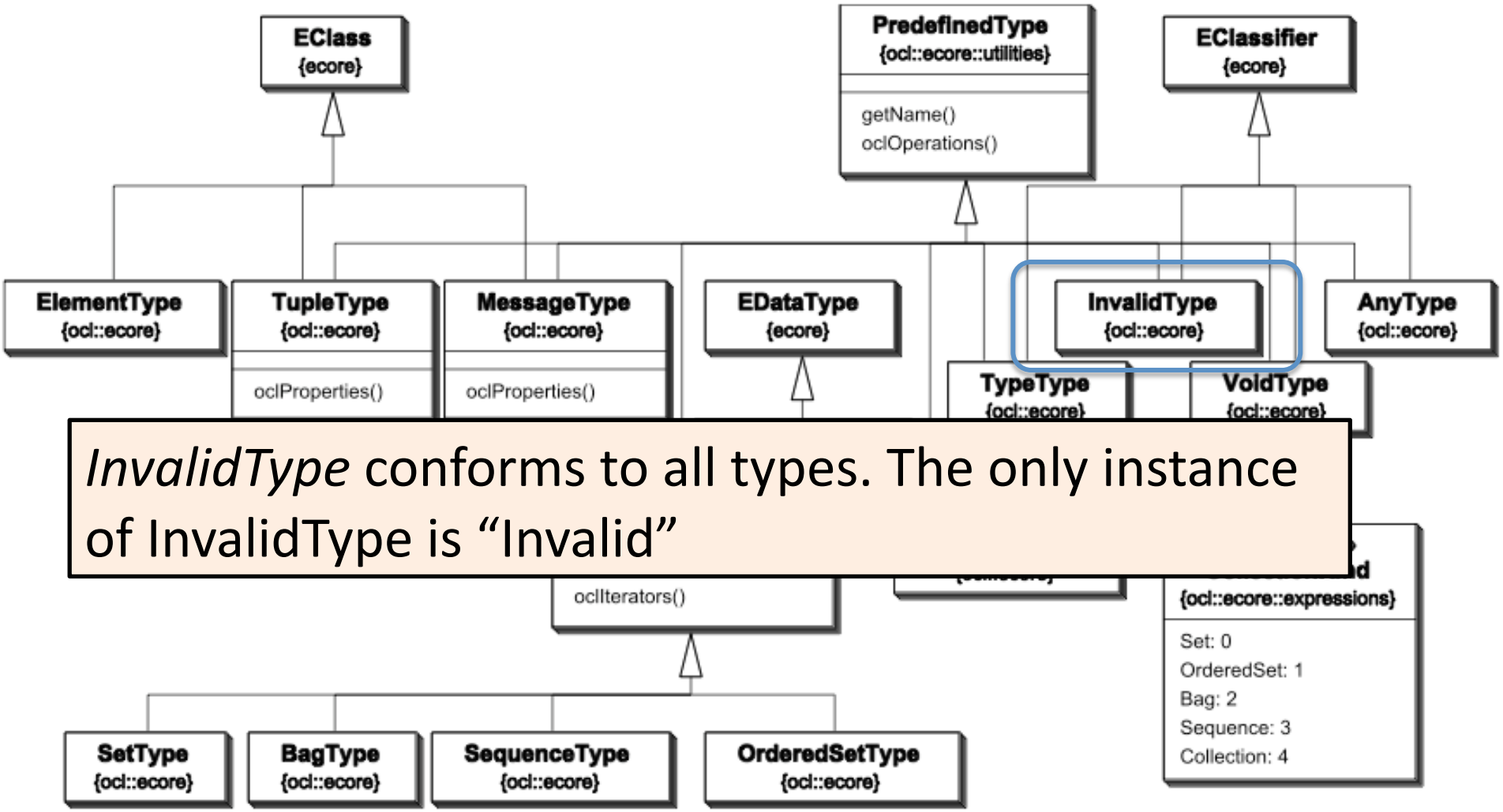
# Basic OCL Types



# Basic OCL Types

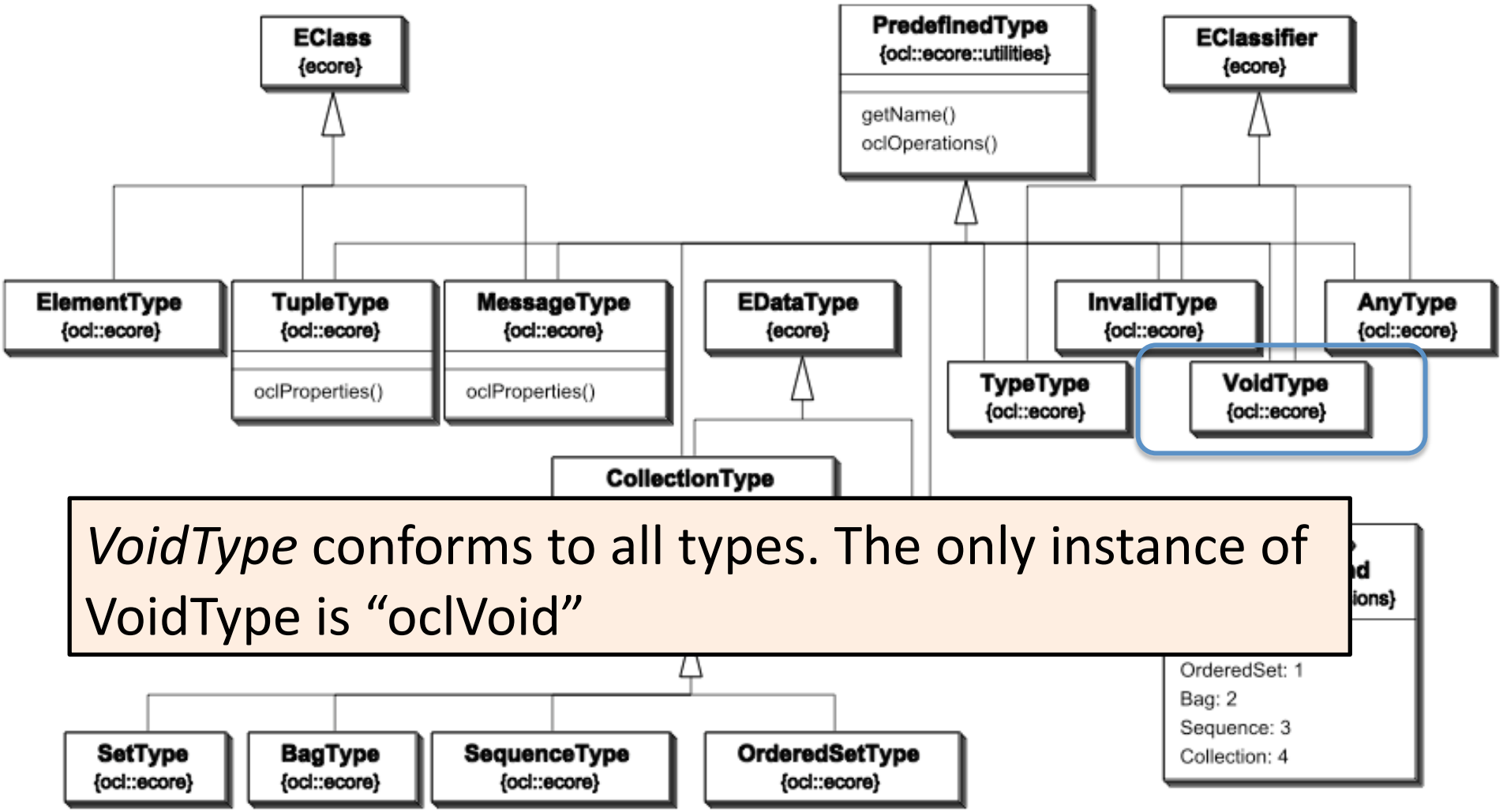


# Basic OCL Types



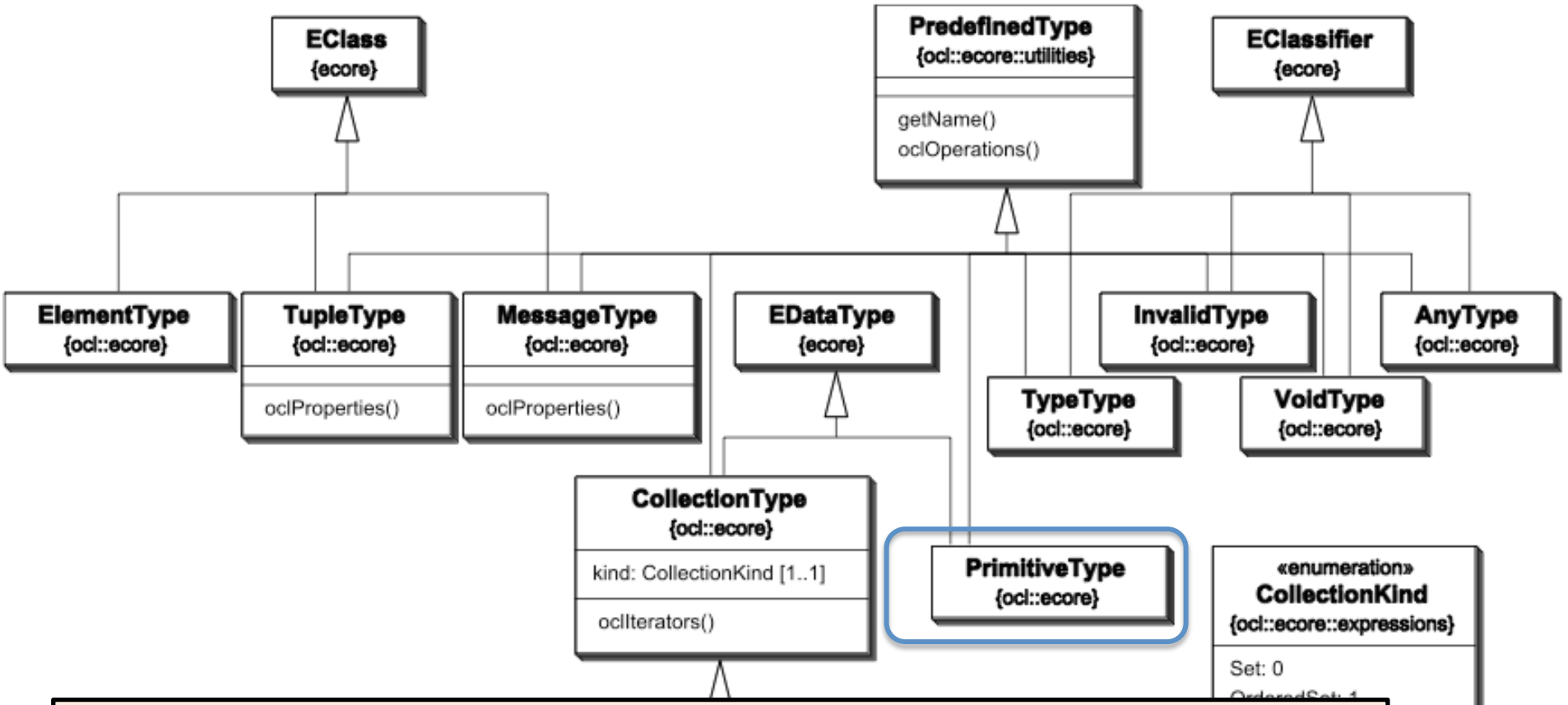

*InvalidType* conforms to all types. The only instance of InvalidType is “Invalid”

# Basic OCL Types



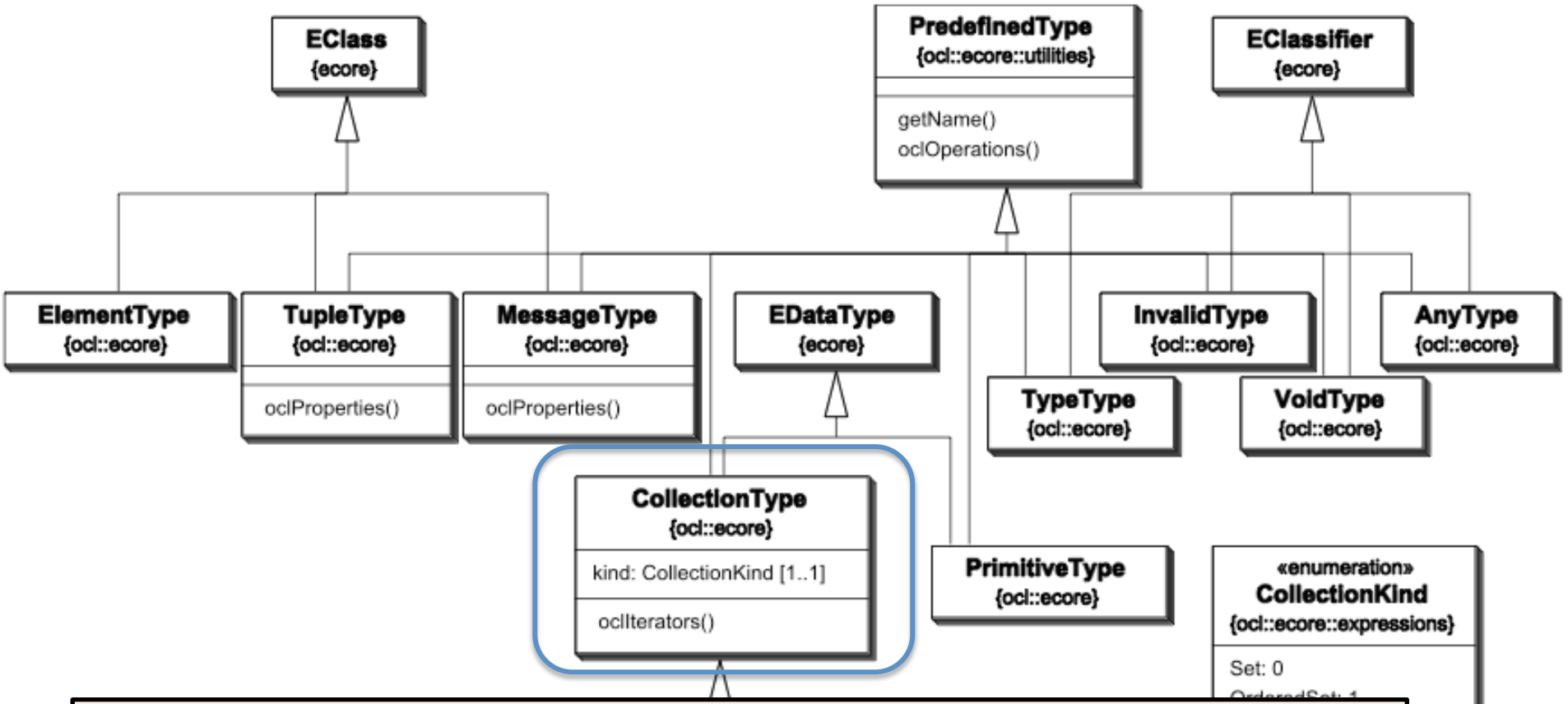

*VoidType* conforms to all types. The only instance of VoidType is “oclVoid”

# Basic OCL Types



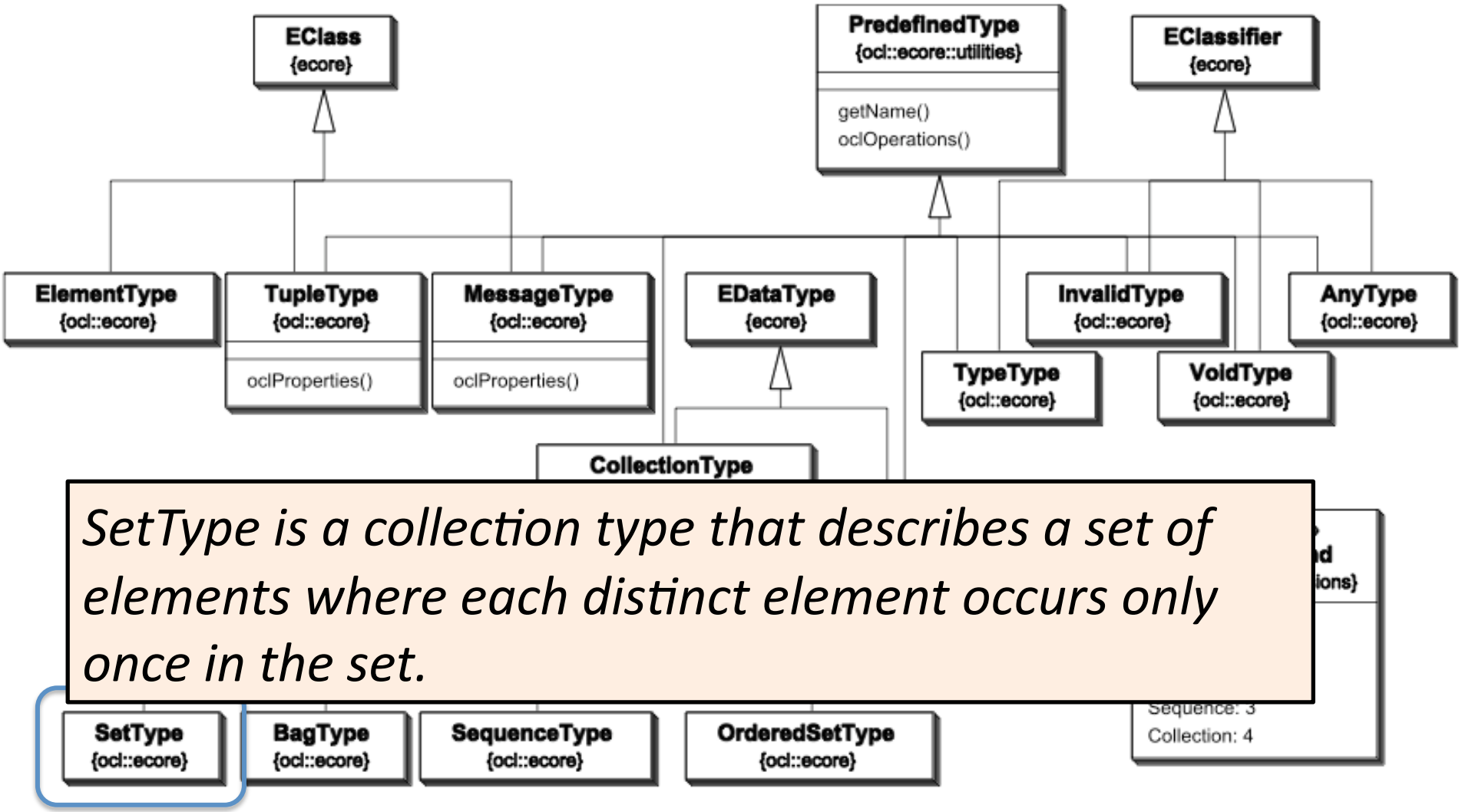

*PrimitiveTypes* in ecore:ocl are EBoolean, EString, Efloat, and EInt

# Basic OCL Types



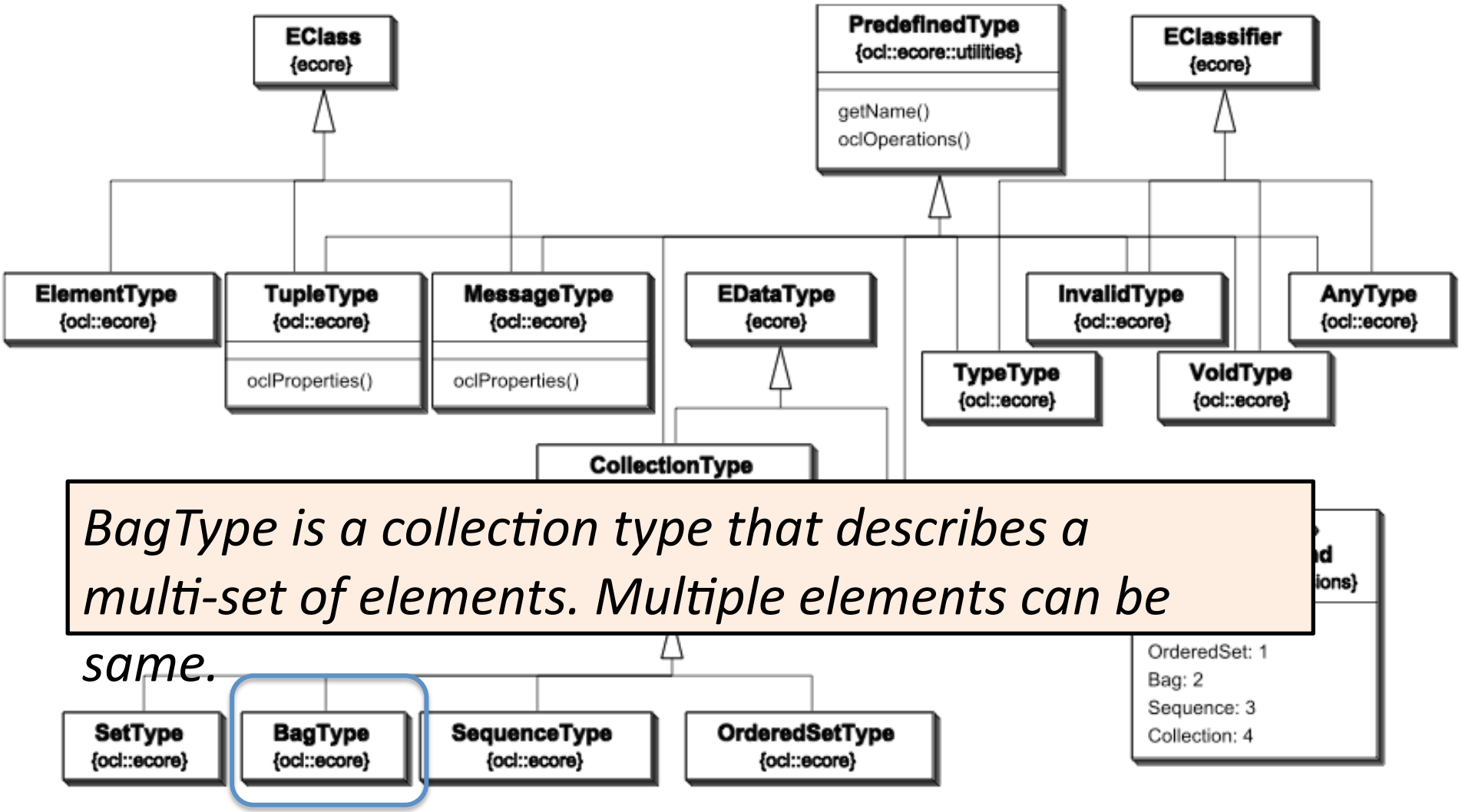

*CollectionType* describes a *list of elements* of a particular given type

# Basic OCL Types

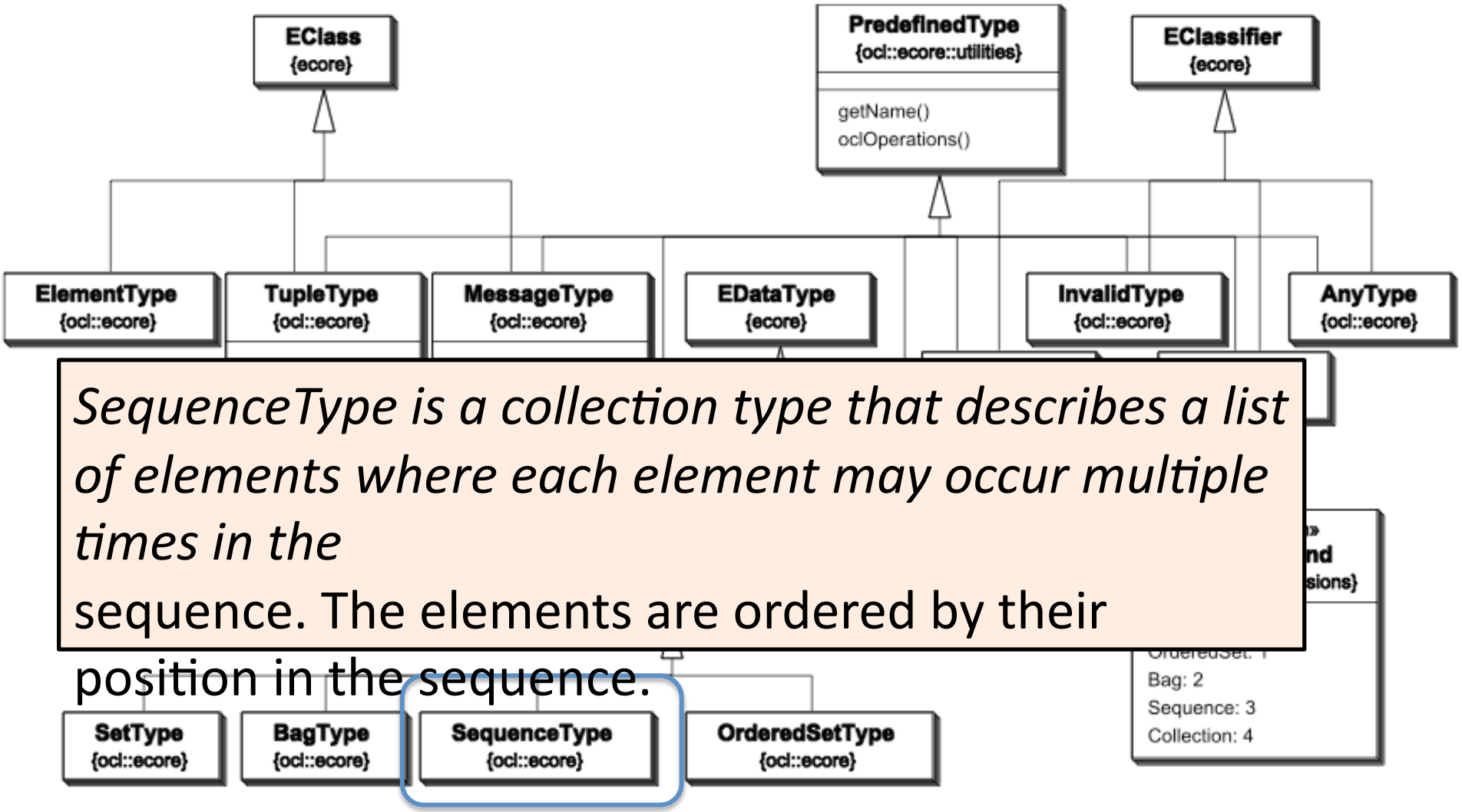



*SetType is a collection type that describes a set of elements where each distinct element occurs only once in the set.*

# Basic OCL Types

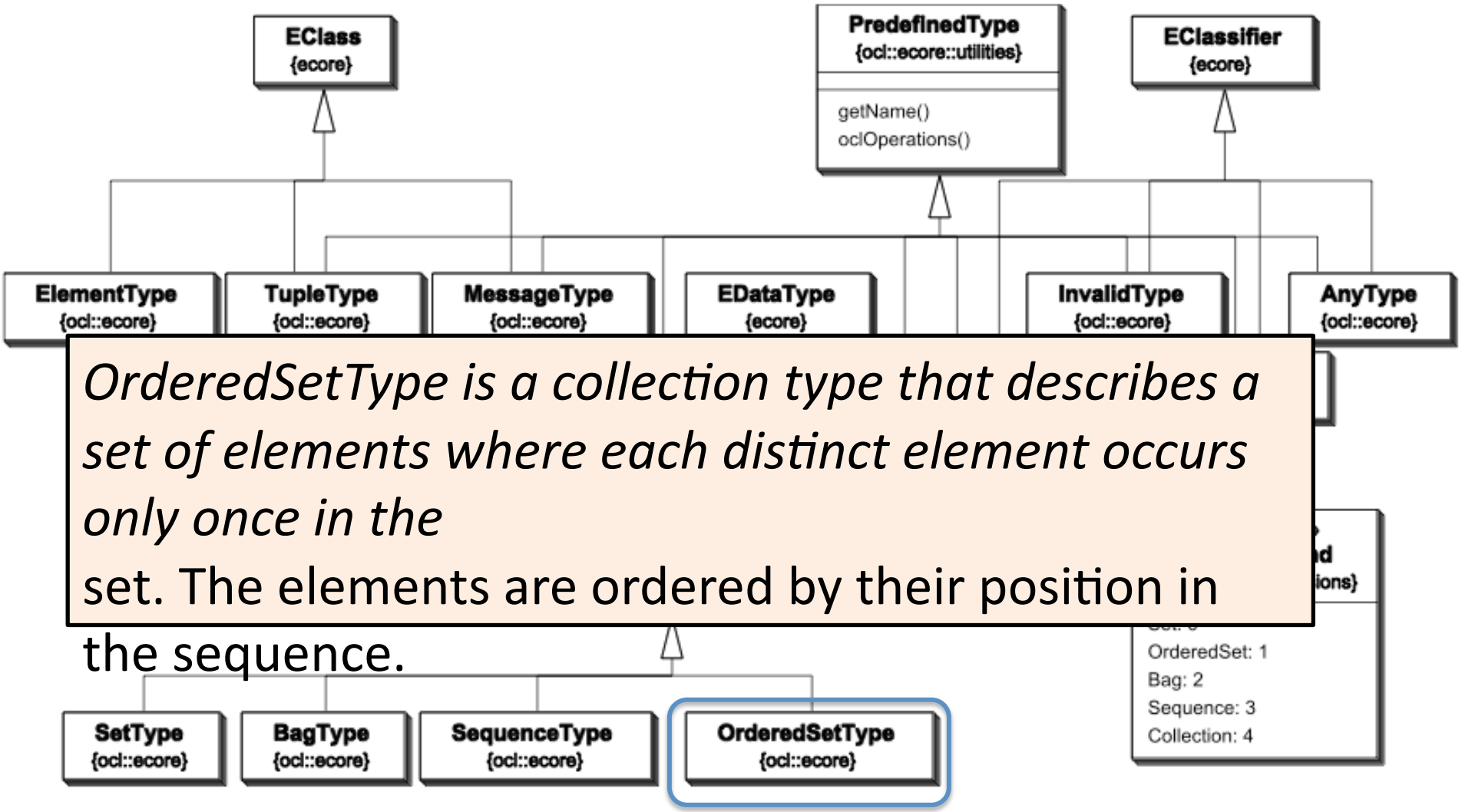


# Basic OCL Types



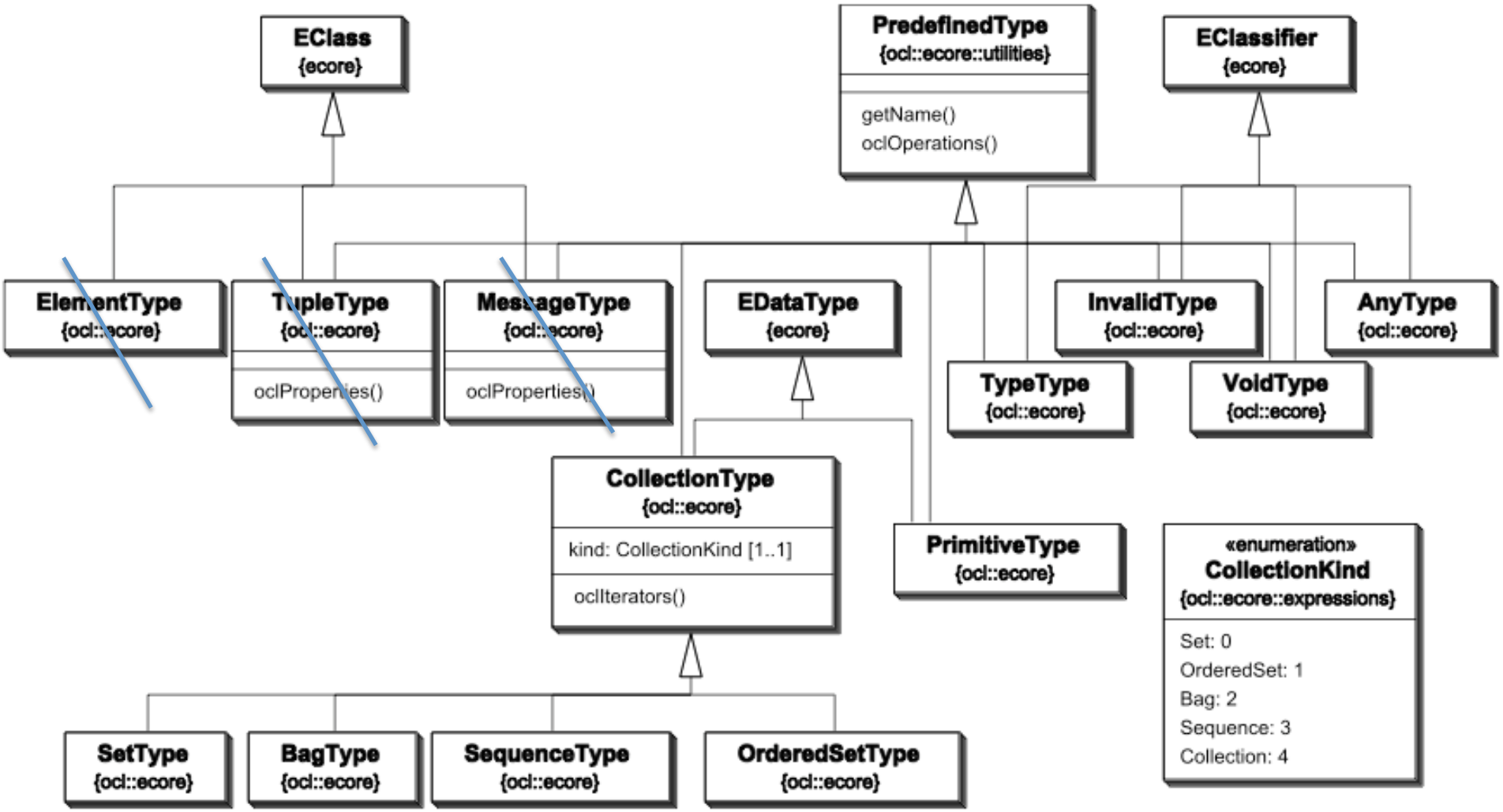

*SequenceType is a collection type that describes a list of elements where each element may occur multiple times in the*
sequence. The elements are ordered by their position in the sequence.

# Basic OCL Types

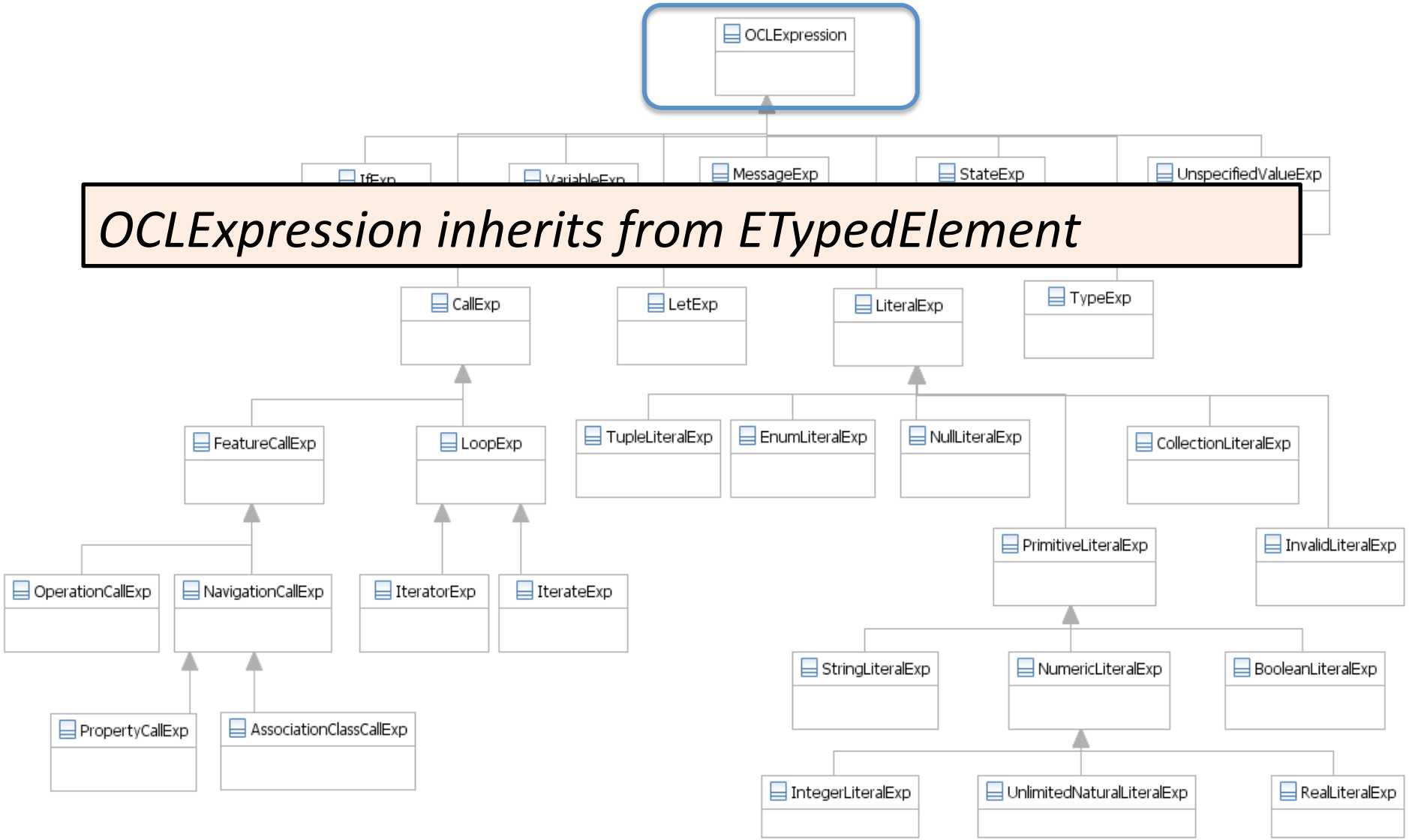



*OrderedSetType is a collection type that describes a set of elements where each distinct element occurs only once in the*
set. The elements are ordered by their position in the sequence.

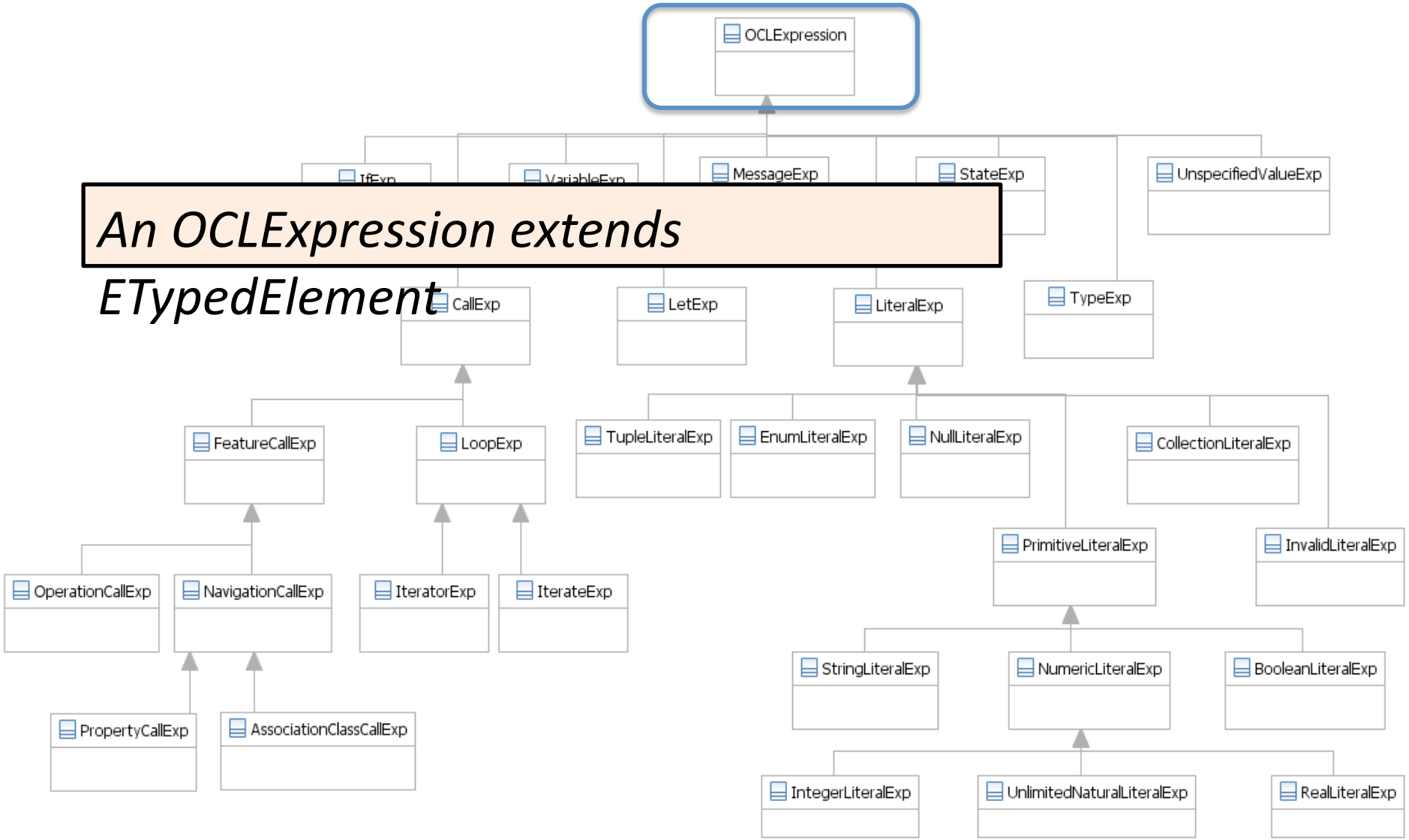
# Not Considered OCL Types



# Basic OCL Expressions

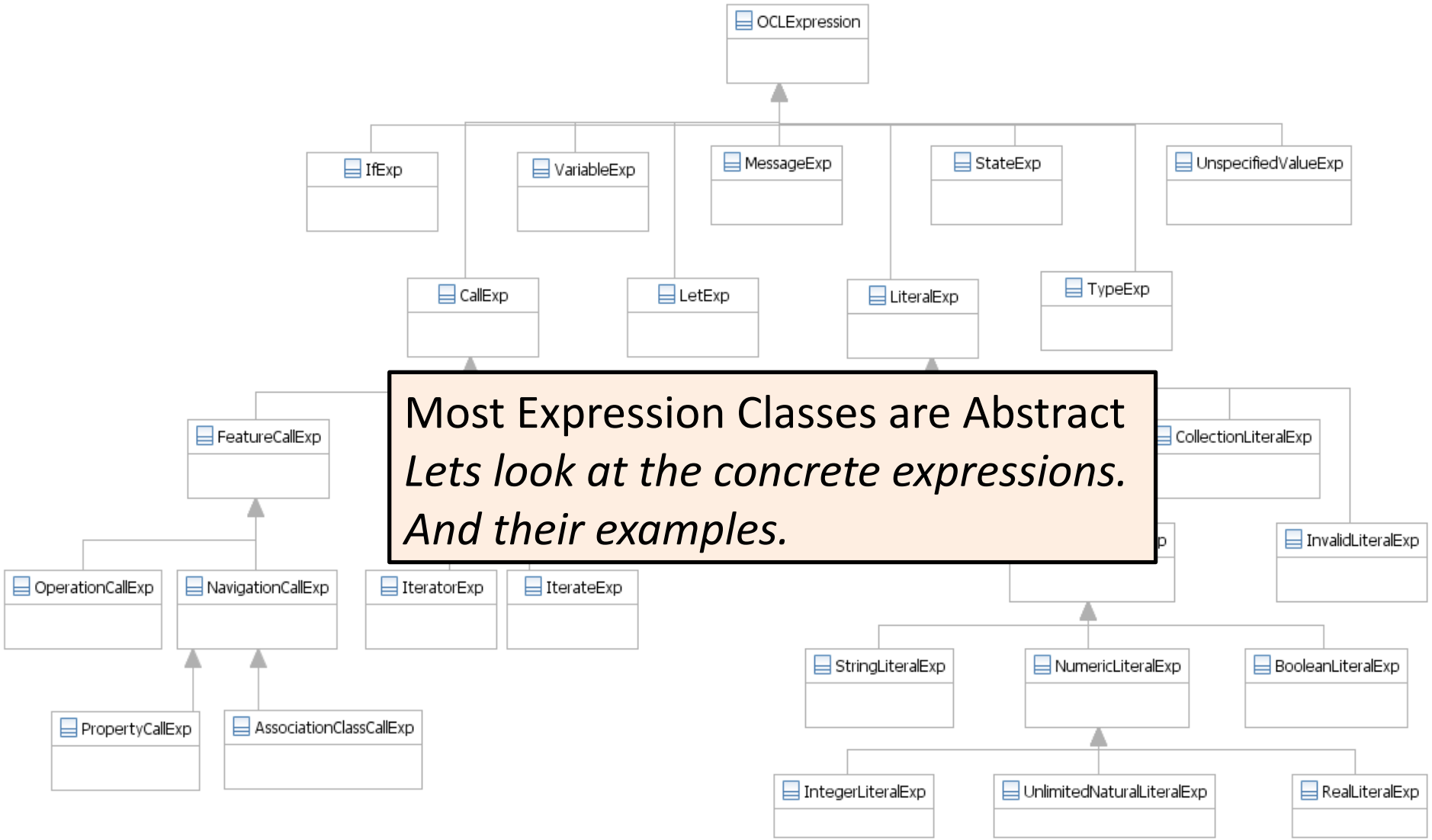


# Basic OCL Expressions




*An OCLExpression extends ETypedElement*

# Basic OCL Expressions

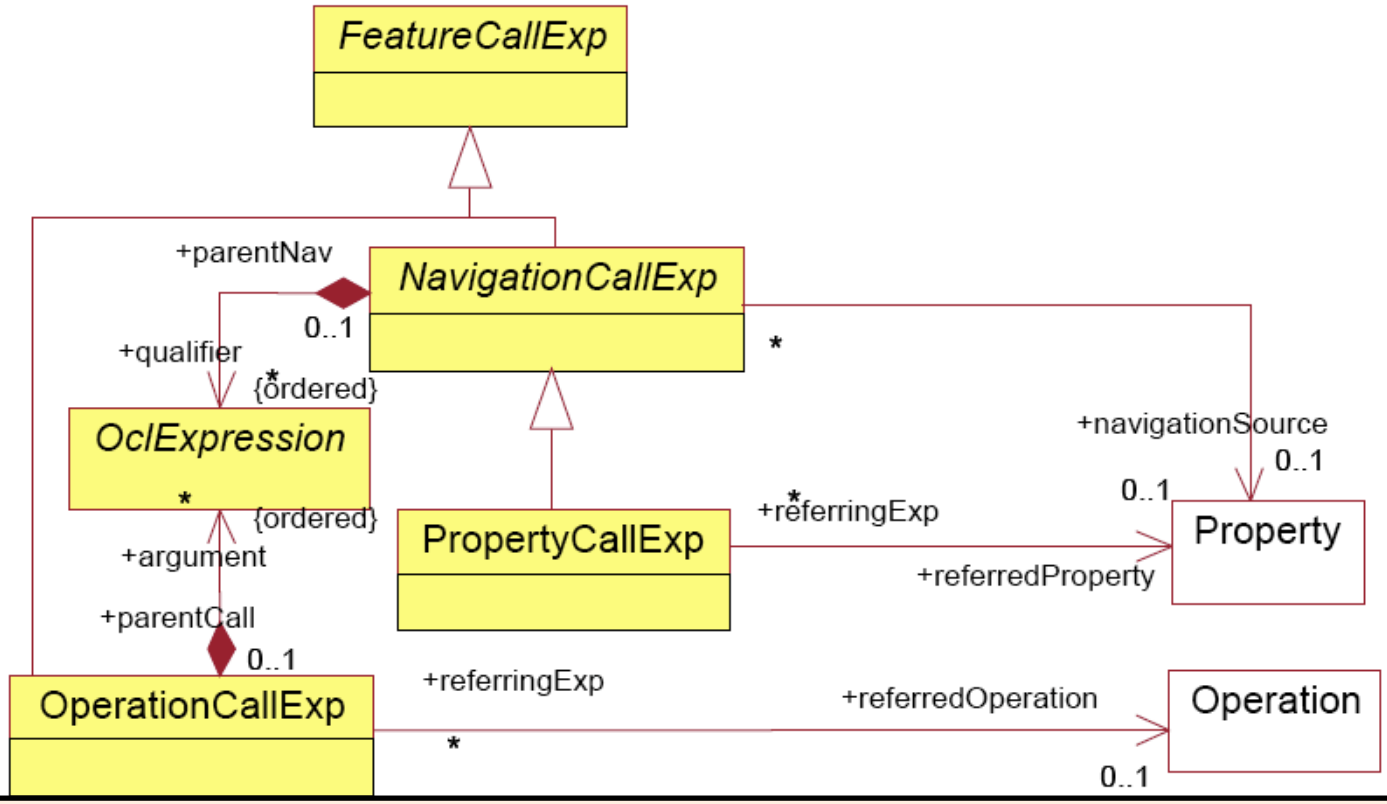



Most Expression Classes are Abstract

*Lets look at the concrete expressions.*

*And their examples.*

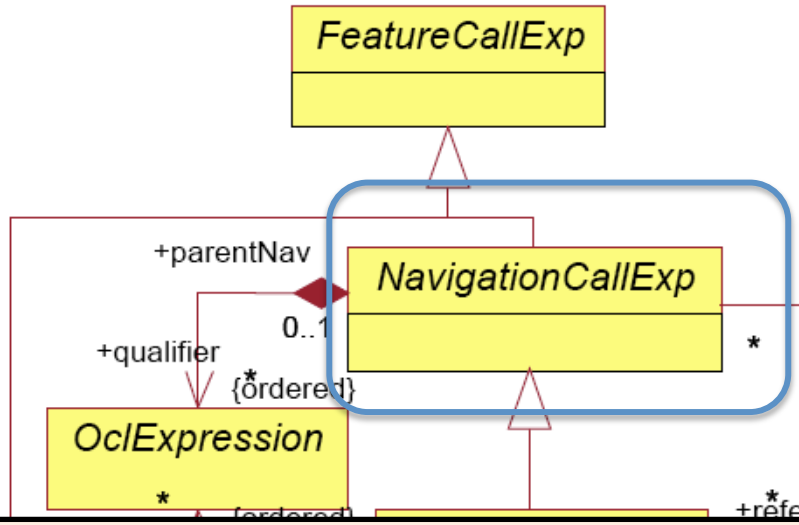
# Basic OCL Expressions




Lets look at Feature Call Expressions. We are interested in

1. *Navigation Call Expressions*
2. *Property Call Expressions*
3. *Operation Call Expressions*

# Basic OCL Expressions



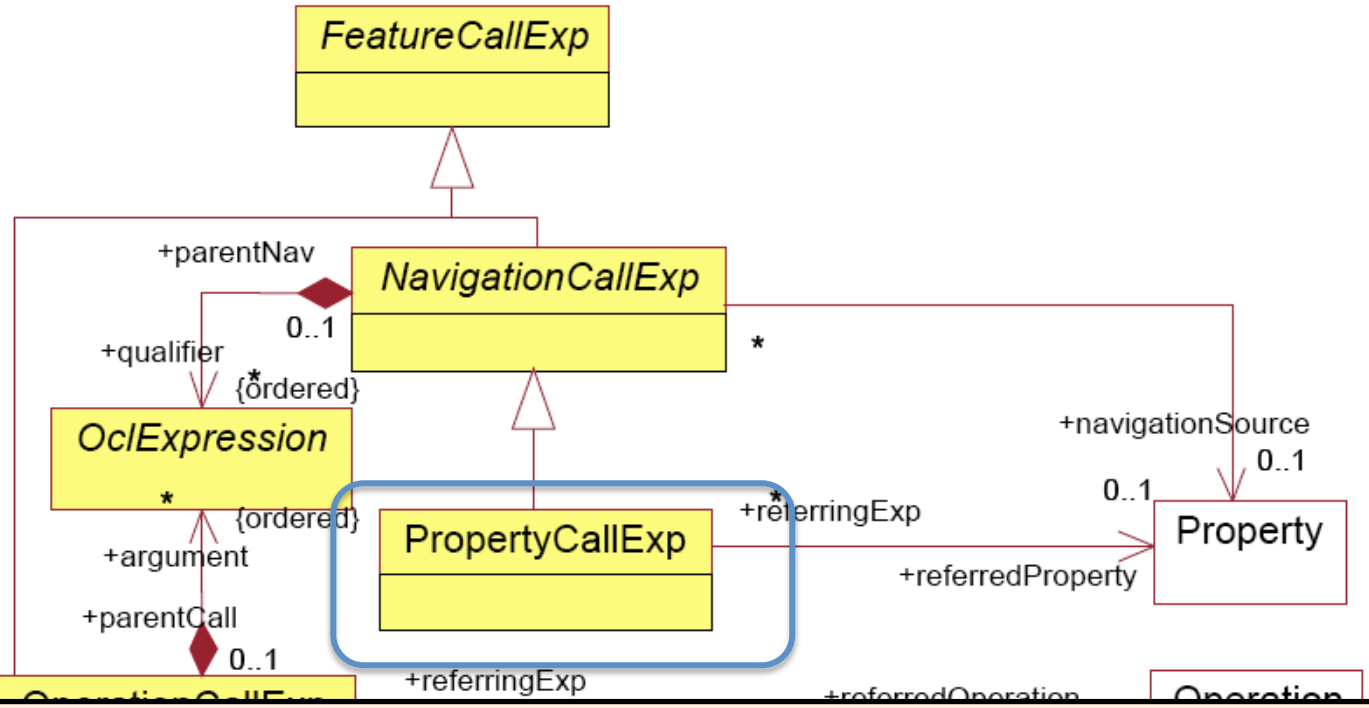

A ***NavigationCallExp*** *is a reference to an AssociationEnd in an ECore model. It is used to* determine objects linked to a target object by an association.

*Eg:*

*context RoadSegment inv:*

*self.RoadSegmentOutPort-->size()=1*

# Basic OCL Expressions



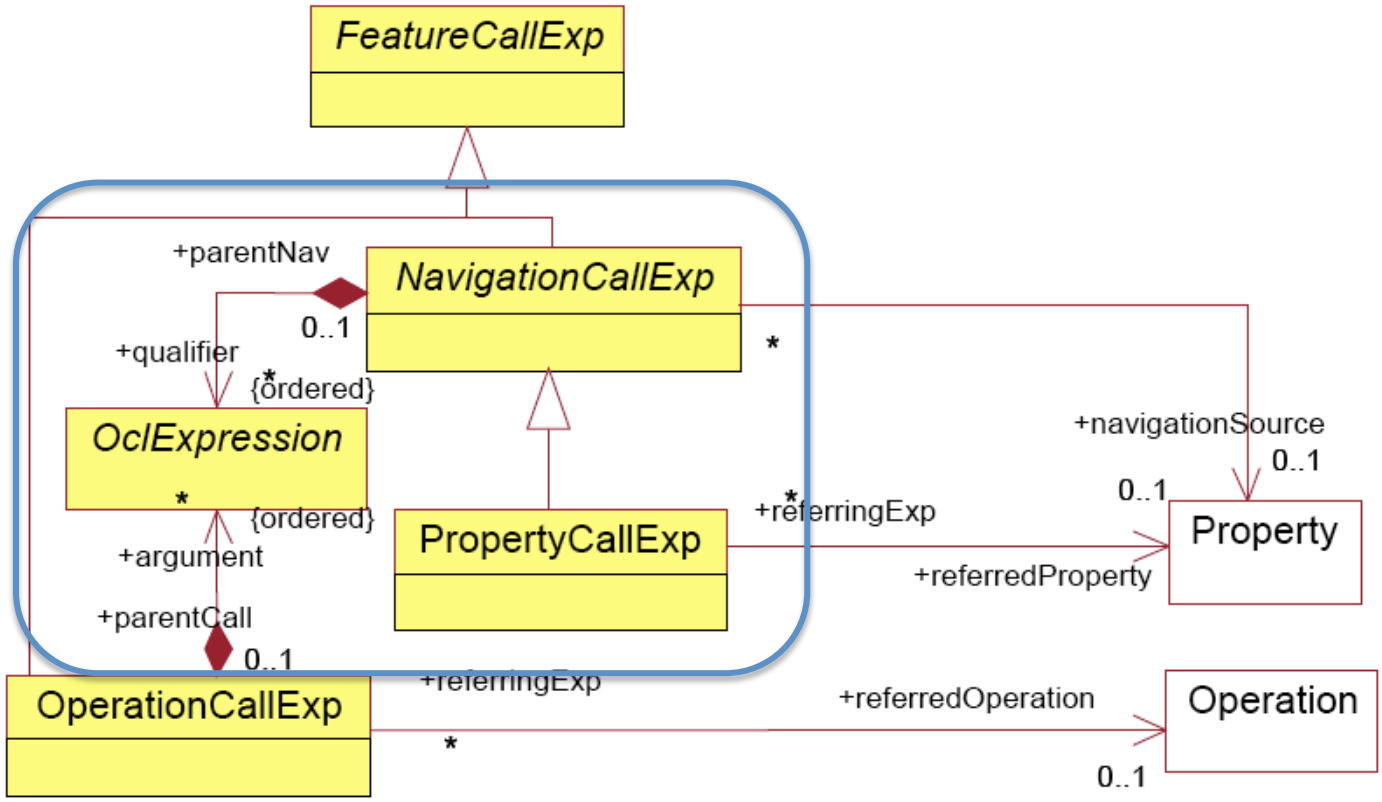

A ***PropertyCallExp*** *is a reference to an Attribute of an EClassifier. It evaluates to the value* of the attribute.

*Eg:*

*context RoadSegment inv:*

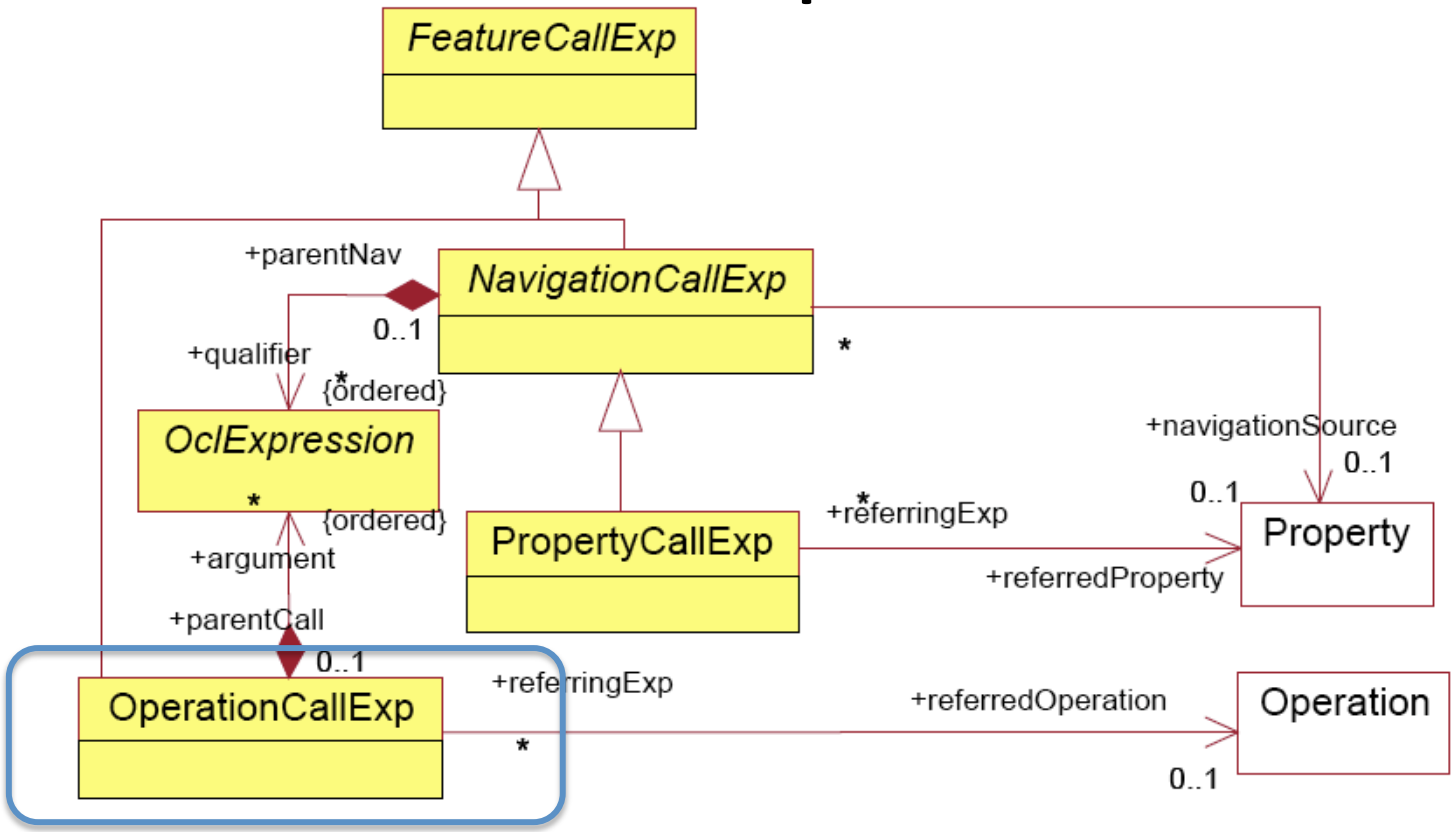
*self.CarCapacity =1*

# Basic OCL Expressions



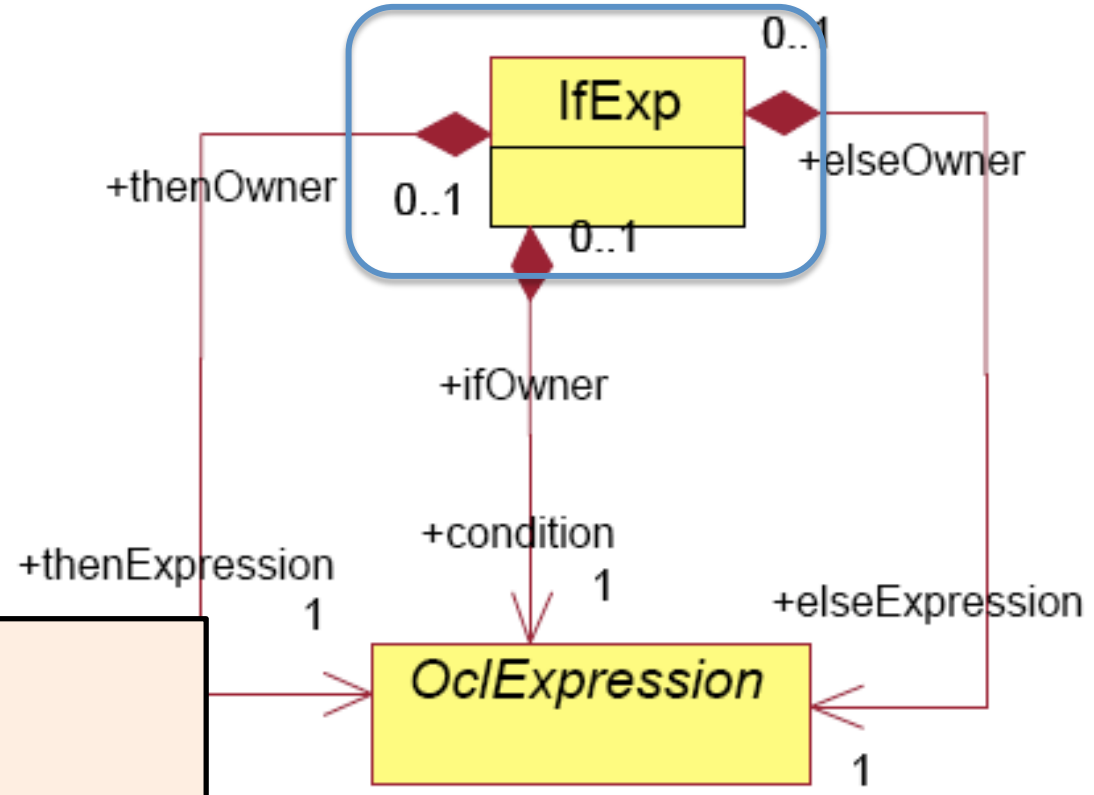

A ***NavigationCallExp*** *and* ***ProperyCallExp*** *can contain 0..* sub OclExpressions*

# Basic OCL Expressions




A ***OperationCallExp*** refers to an operation defined in a Classifier. The expression may contain a list of argument
expressions if the operation is defined to have parameters.
*Eg: +,/,-,concat() etc. for primitive types and sum(), size() for collections*

# Basic OCL Expressions

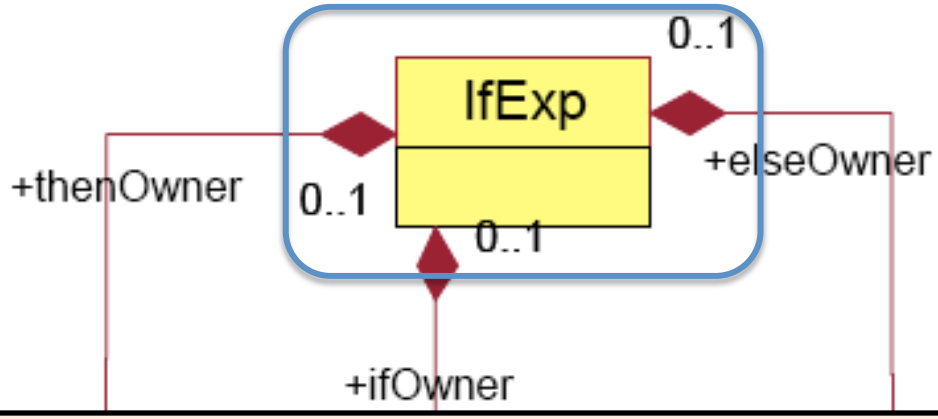



OCL If Expressions

***Contains***

1. *An condition OCL Expression*
2. *A then OCL Expression*
3. *An else OCL Expression*

# Basic OCL Expressions

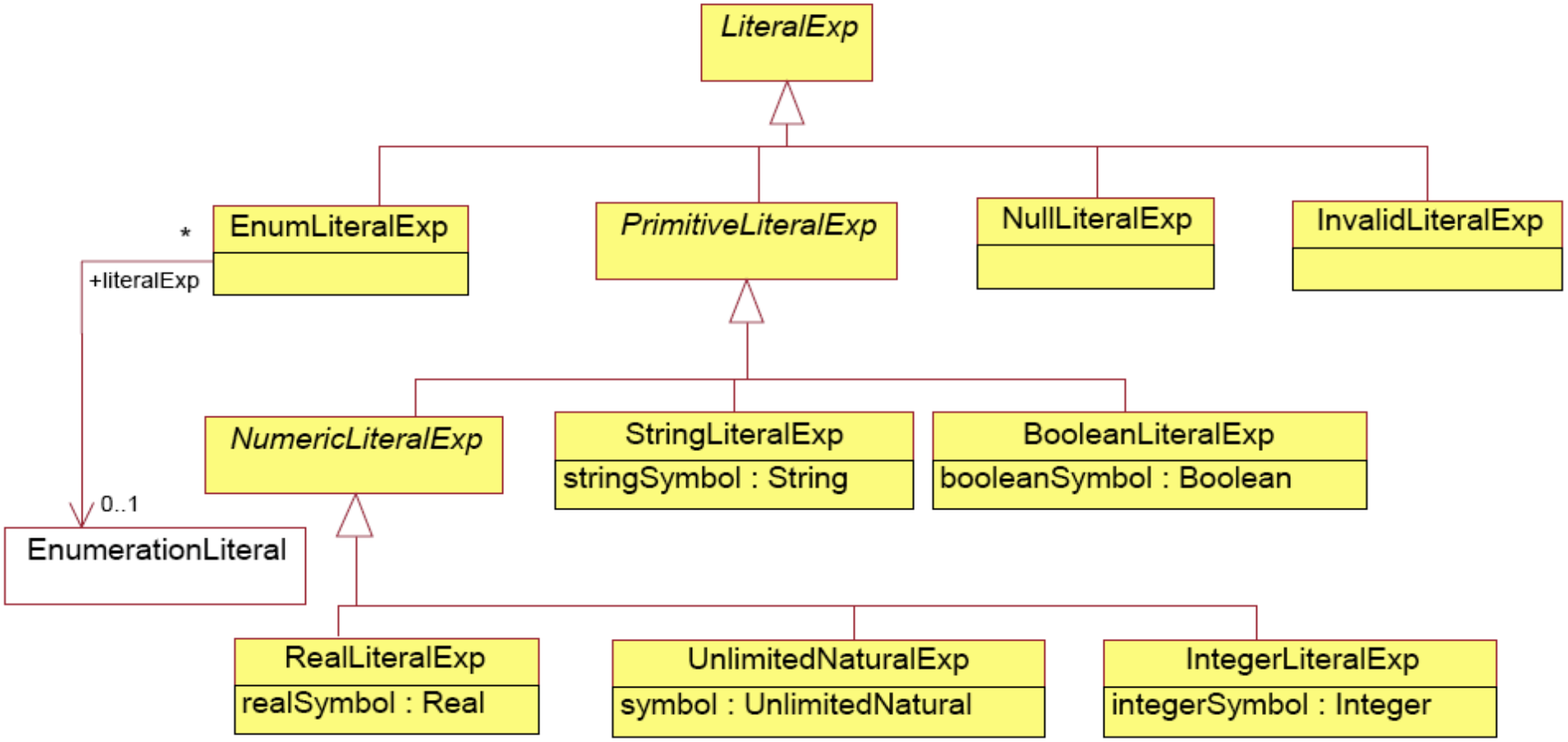



An ***IfExp*** *results in one of two alternative expressions (then/else) depending on the evaluated value of a condition.*

*Eg:*

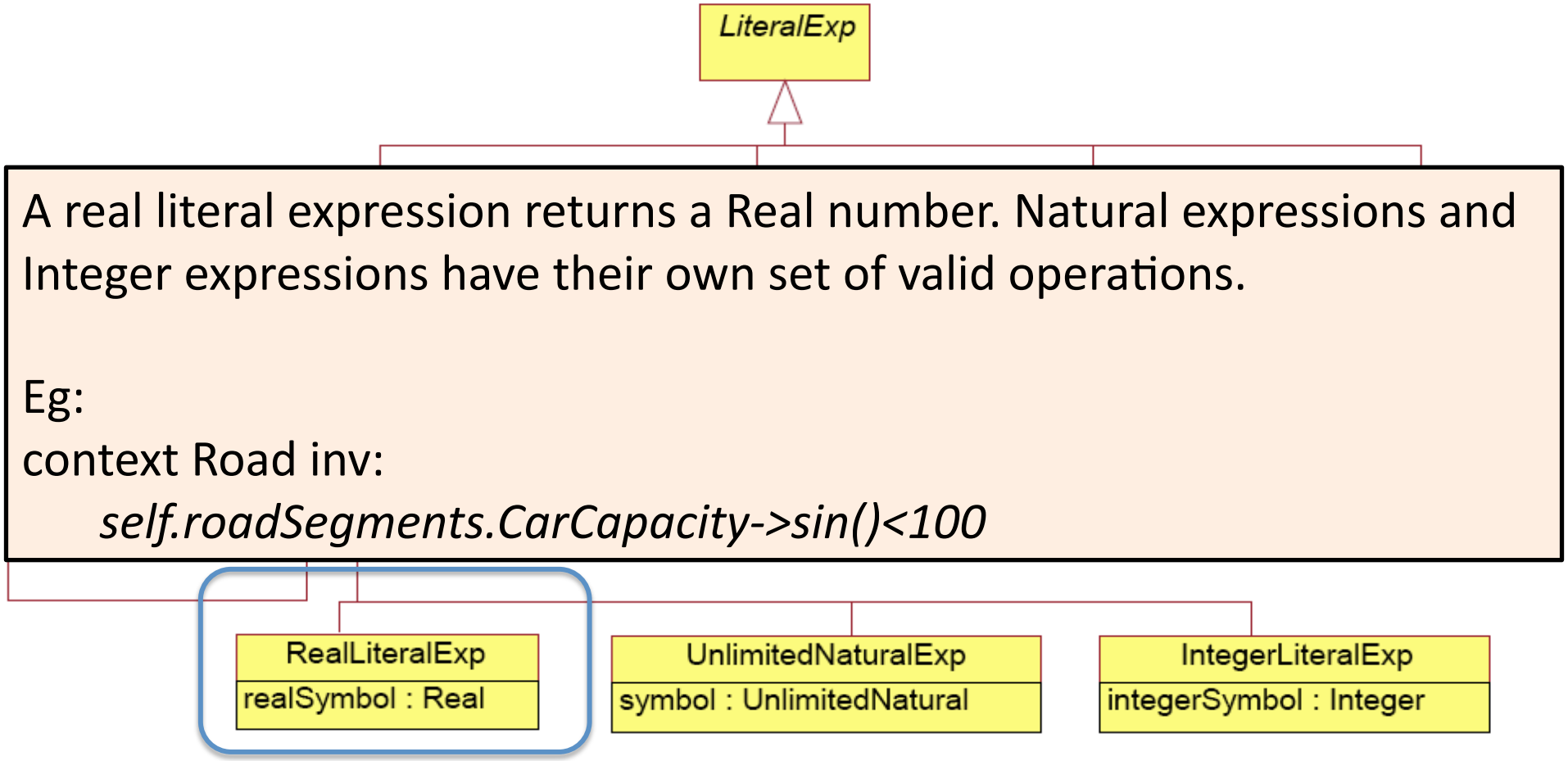
```
context RoadSegment inv:
    if self.CarCapacity > 10 then:
        self.name =“TrafficJamPoint”
    else:
        self.name=“NormalPoint”
```

# Basic OCL Expressions

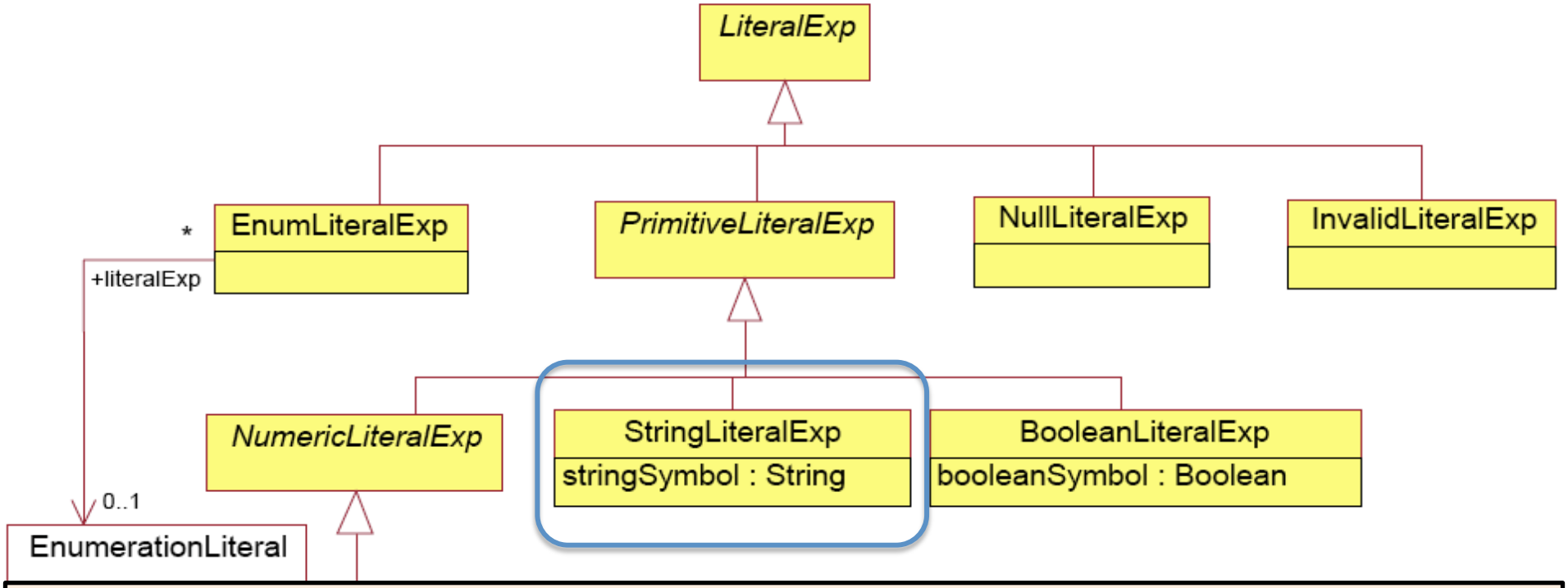



A **LiteralExp** results in a Literal. Eg: The answer can be an EString, *EBoolean, EFloat, or an EInt*

# Basic OCL Expressions



# Basic OCL Expressions



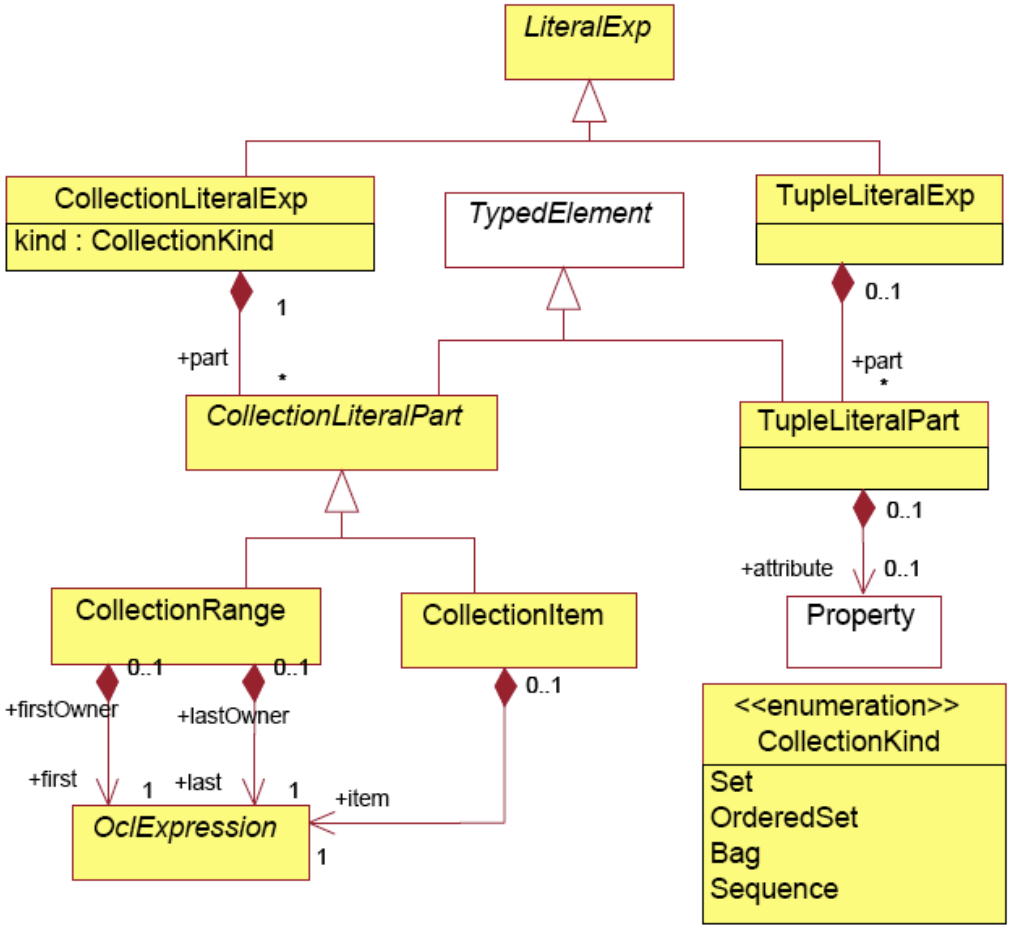

A string literal expression returns a String.

Eg:

context Road inv:

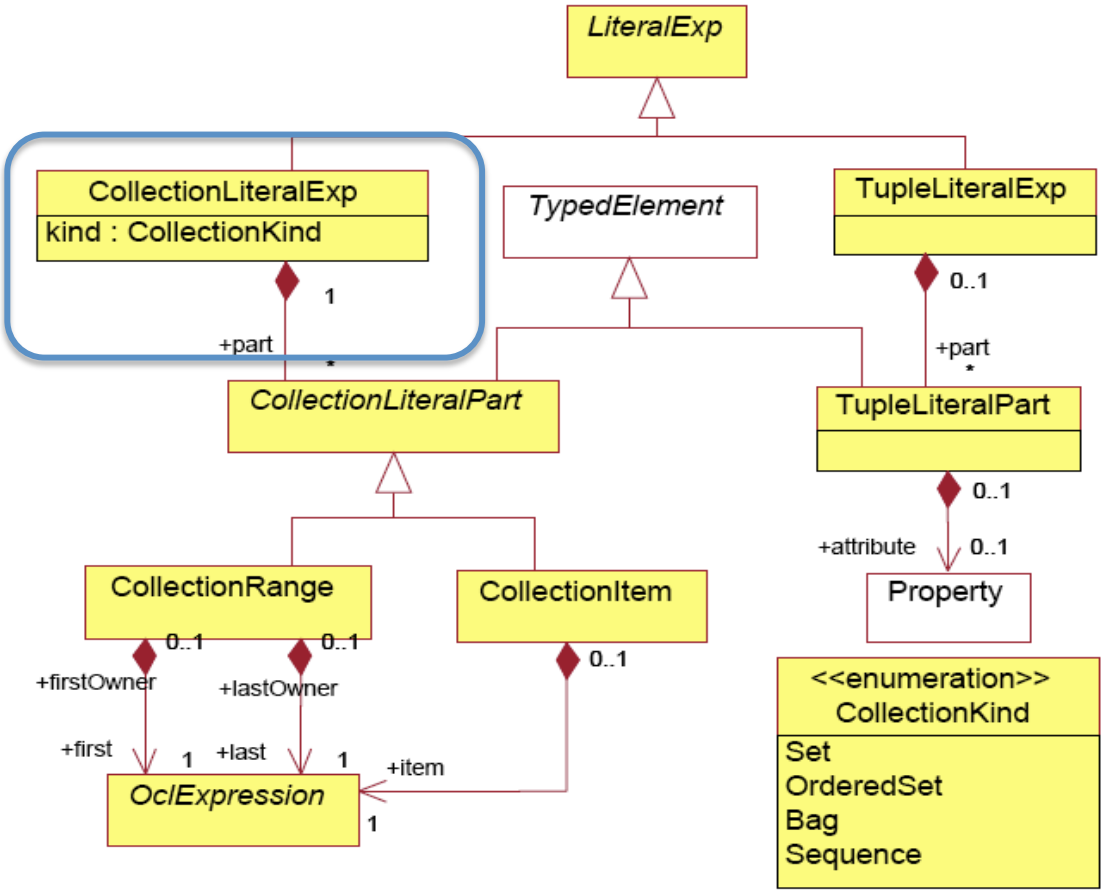
*self.roadSegments.name="DeadEnd"*

# Basic OCL Expressions




A **LiteralExp** also results in collections.

# Basic OCL Expressions




A **LiteralExp** also results in collections. A Collection literal is either one of the Collection Kinds. It has many parts consisting of CollectionItem and Collection Ranges.

Eg: *context r:RoadSegment inv:*

*r.OutPort.RoadConnection->select(Inport|*

*Inport.RoadSegmentInport.RoadSegment.name=r.name)->isEmpty()=True*

# OCL Standard Library: Primitive Types

The OCL standard library is defined from instances of the Primitive Types in the OCL meta-model

OCL has four basic primitive datatypes

* Boolean (true, false) -> EBoolean
* Integer (1, -5, 2, 34, 26524, ...) -> EInt
* Real (with the values 1.5, 3.14, ...) -> EFloat
* String (`To be or not to be...`) -> EString

Operators on Primitive Types:

* Integer has the operations: *, +, -, /, div(), abs(), mod(), max(), min(), sum(), sin(), cos()
* Real has the operations: *, +, -, /, floor(), sum(), sin(), cos()
* Boolean has the operations: and, or, xor, not, implies, if-then-else
* String has the operations: concat(), size(), substring(), toInteger() and toReal()

# OCL Standard Library: Collections

The OCL standard library is defined from instances of the Collection Types in the OCL meta-model

Collection is the abstract superclass of Set, OrderedSet, Bag and Sequence.

* **Set** is a collection without duplicates. Set has no order.
* **OrderedSet** is a collection without duplicates. OrderedSet has an order.
* **Bag** is a collection in which duplicates are allowed. Bag has no order.
* **Sequence** is a collection in which duplicates are allowed. Sequence has an order.

Collection operations

* The number of elements in the collection self: **size()**
* The information of whether an object is part of a collection: **includes()**
* The information of whether an object isn't part of a collection**: excludes()**
* The number of times that object occurs in the collection **self. count()**
* The information of whether all objects of a given collection are part of a specific collection**: includesAll()**
* The information of whether none of the objects of a given collection are part of a specific collection: **excludesAll()**
* The information if a collection is empty**: isEmpty()**
* The information if a collection is not empty: **notEmpty()**

# OCL Standard Library: Collections

**Iterators over collections**

* The selection of a sub-collection: **select()**
* When specifying a collection which is derived from some other collection, but which contains different objects from the original collection (i.e., it is not a sub-collection) use: **collect()**
* The information of whether an expression is true for all objects of a given collection**: forAll()**
* The addition of all elements of a collection: **sum()** Elements must be of a type supporting the **+** operation.

# Testing OCL Expressions

- Test-cases for all Primitive Type Operations for all types
- Test cases for Navigation Expressions
- Test cases for Property Expressions
- Test cases for all Collection Operations
- Test cases for all Iterators over Collection Operations

THANKS