

Object Constraint Language

1

What is a Constraint?

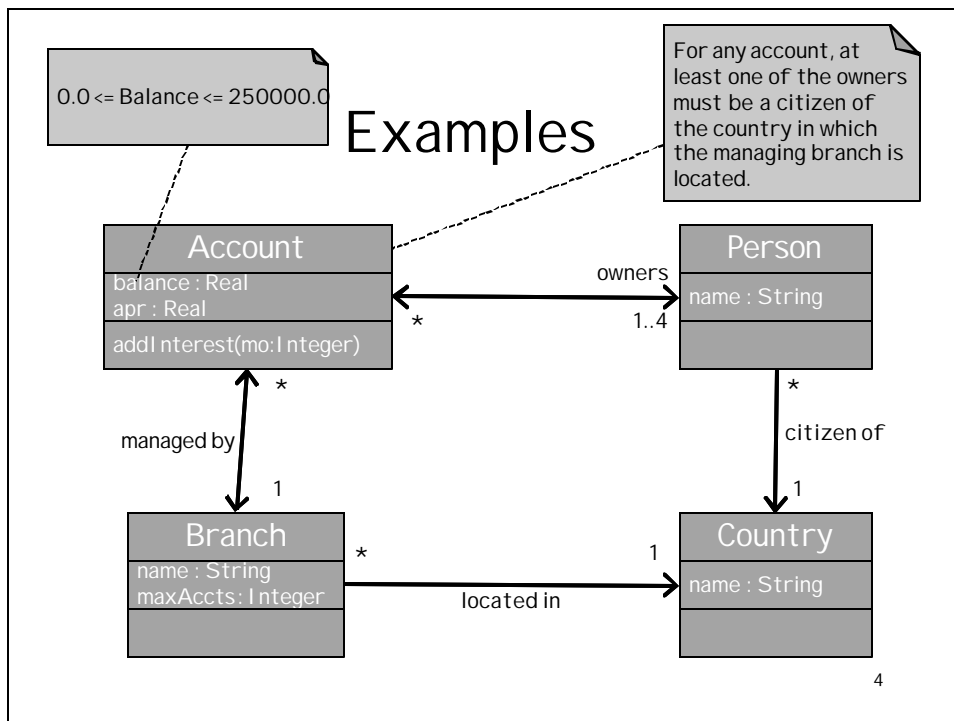
- A restriction on information:
 - Attribute values.
 - Links.
 - Results of query operations.
 - A query operation is one that has no side effects - i.e., doesn't change any attribute values.
 - Preconditions and post conditions of operations.

2

Expressing Constraints

- A constraint is expressed as a Boolean expression (i.e., has a true or false value).
- Choices of notation:
 - English (or other natural language).
 - OCL.

3



Examples Expressed in OCL

- context: Account
 - inv balanceLimit : self.balance >= 0.0 and self.balance <= 250000.0
 - inv ownershipRestriction : self.owners -> exists (o : Person | o.country = self.branch.country)

5

OCL Syntax -- Context

- The first line in any OCL constraint is the context.
- It usually is the name of a classifier.
- It indicates the starting point for references to information being constrained.
- The keyword *self* refers to an instance of the context.
- The dot operator may be used to refer to properties of the context object (attributes, operations or links).

6

OCL Syntax -- Constraints

- After the context, there may be any number of constraints.
- Each constraint has the following syntax:
 - <constraintType> <constraintName>:
 <Boolean expression>
 - The constraintName is optional.

7

Constraint Types

- inv – Invariant
 - All instances of the context class must satisfy the invariant.
 - When they are constructed.
 - Whenever their state is changed.
 - Includes attribute values as well as links.
- pre – Precondition of an operation.
 - Assumed to be true before the operation is executed.
- post – Post condition of an operation.
 - Guaranteed to be true after an operation executes
- (there are others – see the reference manual)

8

No Side Effects

- The constraint has no side effects.
- That is, it doesn't change the values of any data (attributes or links).
- This especially applies to post conditions.

9

Subclasses

- A subclass invariant is "anded" to the superclass invariant.
 - In this way, subclasses may only have more restrictive invariants than their parents.
- The post condition on an overridden operation is "anded" to the post condition of its parent.
 - Thus it is only stronger than its parent.
- The precondition on an overridden operation overrides that of its parent, and may only be weaker than the precondition on its parent.
- These rules are because of the Liskov substitutability principle (an instance of a subclass should be able to stand in for an instance of its superclass).

10

Behavior Specifications

- To specify the behavior of an operation, you would specify preconditions and post conditions.
- To help, you can write the post condition in terms of the final values of attributes or links in terms of the original (precondition) values of attributes and links (use the `@pre` operator).
- To show returned values, use the *result* keyword.

11

Using *@pre* and *result*

- context Account
post addInterest(mo:Integer):
 balance = balance@pre * (1+ apr * mo/12.0)
post computeInterest(mo:Integer):
 result = balance*apr * mo/12.0

Account
balance : Real
apr : Real
addInterest(mo:Integer)
computeInterest(mo:Integer) : Real

12

OCL Collections

Name	Ordered?	Duplicate elements allowed?
Set	No	No
Bag	No	Yes
OrderedSet	Yes	No
Sequence	Yes	Yes

13

Use of OCL Collection Types

- You may use OCL collection types when declaring attributes or operation signatures.
- Examples:
 - owners : Set(Person)
 - accountList: OrderedSet(Account)
 - history : Sequence(Transaction)
 - getAccountsFor(p:Person):Set(Account)

14

Navigation with the dot operator

- Starting with the *self* object, you may navigate to other objects that might be linked to it using the dot operator.
 - You may refer to the linked object either by its rolename (if shown on the association), or by its class name (with the first letter in lower case).
 - If the upper bound of the multiplicity of the referenced class is 1, the dot operator results in an instance of the referenced class.
 - If it is >1, the dot operator results in a Set of objects of the referenced class.

15

Other Ways To Navigate

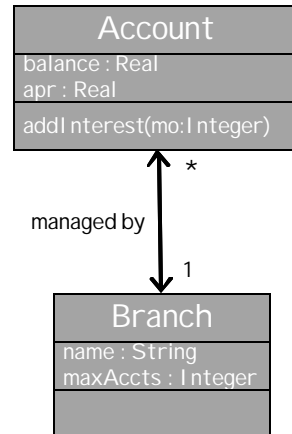
- The following may also be used to navigate to other objects in an OCL expression:
 - A query operation that returns a reference to an instance of (or a collection of instances of) a linked class.
 - An attribute whose value is a reference (or collection of references) to a linked class.

16

Collection Operators

- To apply an operation to a collection, you must use the arrow operator (->)
 - Example:


```
context Branch
inv: self.account->size() <
maxAccts
```



17

Collection Operators

Operator	Examples	Results
Conversion	<code>c->asBag()</code>	Bag
Query	<code>c->isEmpty()</code> <code>c->size()</code>	Boolean Integer
Access	<code>c->first()</code>	Account
Selection	<code>c->append(x)</code>	Sequence
Boolean Iterator	<code>c->exists(i:Account i.balance=0.0)</code> <code>c->forall(i:Account i.balance<100.0)</code>	Boolean Boolean
Selection Iterator	<code>c->any(i:Account i.balance<50.0)</code>	Account

Note 1: In this example:

- `c` is an Sequence of Account objects.
- `x` is an object instance of Account.

Note 2: Collections are *immutable*. Thus, `c->append()`, in our example, would create a new Sequence.

18

Example

- A Smokestack is like a Stack. It is last-in-first-out (LIFO), but it has a maximum depth. If the smokestack is full, pushing a new element on will result in the bottommost element being deleted (turned to smoke).
- Operations are push, pop, top, isEmpty, isFull.

19

Smokestack

Smokestack
s:Sequence(Object) = { } maxSize:Integer
<u>Smokestack(m:Integer):</u> Smokestack push(e:Object) pop() top():Object isEmpty():Boolean isFull():Boolean

20

Constraints

```
context Smokestack
inv maximumDepth: s->size() <= maxSize
post Smokestack (m:Integer): maxSize = m -- constructor
post push(e:Object): if s->size()<maxSize
    then s=s@pre->prepend(e)
    else s=(s@pre->subSequence(1,maxSize-1))->prepend(e)
pre pop(): s->notEmpty()
post pop(): s=s@pre->subSequence(2,s@pre->size)
pre top(): s->notEmpty()
post top(): result = s->first()
post isEmpty(): result = s->isEmpty()
post isFull(): result = s->size() = maxSize
```

21