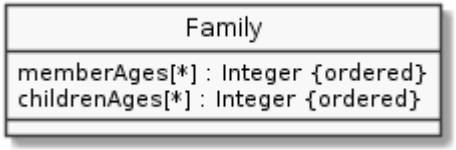
Introduction

- OCL evaluation must be side-effect free
- But constraints may be used otherwise, e.g.:
  a. Generating models satisfying these constraints
     - e.g., test generation
  b. Updating models so that they are satisfied
     - e.g., synchronization of element properties (may be used for derived features, in transformation engines, etc.)
- Achieving b. may rely on **active operations[1]**

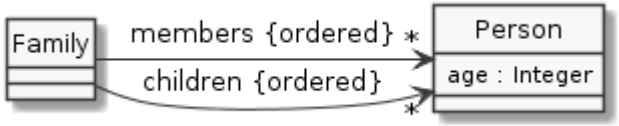
[1] O. Beaudoux *et al.*, *Active Operations on Collections*, MODELS'10

OCL Active Operations

- Basis: incremental algorithms for computing **OCL operations on collections**
- Principle: observation of **mutation events** on source collections
- Example 1:
  - self.childrenAges := self.memberAges->select(e | e < 18)
  - "*For each* e *added into* memberAges *at index* i, *add* e *into* childrenAges *at index* j=(see [1]) *if* e<18."



- Example 2:



  - self.children := self.members->select(e | e.age < 18)
  - In this case, *e.age* can change, therefore we may also need to update *children* if the *age* of any element of *members* changes.
- Algorithms for all OCL collection operations have been specified and implemented.
- Remark: the above examples use a new ":=" operator instead of "=" to denote the direction of updates. An operator denoting bidirectional synchronization also exists.