Πανεπιστήμιο Κρήτης, Τμήμα Επιστήμης Υπολογιστών
Φθινόπωρο 2007

HY351 – Ανάλυση και Σχεδίαση Πληροφοριακών Συστημάτων
Information Systems Analysis and Design

OCL
Object Constraint Language

Διάθρωση

- Άσκηση 1 : Multiplicity constraints using OCL
- Άσκηση 2 : Person marital status
- Άσκηση 3 : Order line constraints
- Άσκηση 4 : Διαχείριση αερογραμμών
- Άσκηση 5 : Text Browser
- Άσκηση 6 : Graph Visualizer

Άσκηση 1 (1/2)

Έστω ότι τα διαγράμματα κλάσεων της UML δεν είχαν συμβολισμούς για την έκφραση περιορισμών αντιστοίχησης (multiplicity constraints) στις συσχετίσεις. Δείξτε πως θα μπορούσαμε να εκφράσουμε αυτούς τους περιορισμούς χρησιμοποιώντας την OCL.

- *Προσπαθήστε να δώσετε παραπάνω από έναν τρόπο έκφρασης για κάθε έναν από τους 4 τύπους αντιστοίχισης.*

Άσκηση 1 (2/2)

[A] -- x----- rel ----- y -- [B]

y:0..n → Δεν χρειάζεται τίποτα

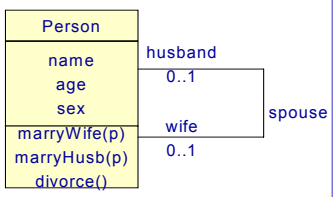
y:1..n →
```
context A inv: not self.rel->isEmpty()
```
ή
```
context A inv: self.rel->Size() >= 1
```

y:0..1 →
```
context A inv: self.rel->Size() <= 1
```

y:1..1 →
```
context A inv: self.rel->Size() == 1
```

Άσκηση 2 (1/4)

Θεωρώντας το διάγραμμα της εικόνας, εκφράστε σε OCL τους ακόλουθους περιορισμούς:



1. A person cannot be married with himself (herself)
2. If x is the wife of y, then x is female
3. If x is the husband of y, then x is male
4. A person can be married only if its age is greater than 18
5. Γράψτε τις προϋποθέσεις (pre-conditions) και τις «μετασυνθήκες» (post-conditions) για τις
   1. `marryWife(p:Person)`
   2. `marryHusband (p:Person)`
   3. `divorce()`

Κάνετε την υπόθεση ότι δεν έχουμε πάντα εξ' αρχής πλήρη γνώση των στοιχείων του κάθε προσώπου. Για παράδειγμα, αν υπάρχουν δύο πρόσωπα p1 και p2 και δεν γνωρίζουμε το φύλο τους, τότε η p1.marryWife(p2) συν τοις άλλοις θα συμπληρώσει κατάλληλα τις πληροφορίες φύλλου. Υπάρχει όμως η περίπτωση να ξέρουμε το φύλο του ενός ή και των δύο.

Άσκηση 2 (2/4)

- A person cannot be married with himself (herself)

```
context  Person
inv:(self.wife->notEmpty() implies not (self.wife == self)) and
     (self.husband->notEmpty() implies not (self.husband == self))
```

- If x is the wife of y, then x is female

```
context Person
inv: not(self.wife->isEmpty()) implies (self.wife.sex == "female")
```

- If x is the husband of y, then x is male

```
context  Person
inv: not(self.husband->isEmpty()) implies (self.husband.sex == "male")
```

- A person can be married only if its age is greater than 18

```
context Person
inv: (self.wife->notEmpty() or self.husband->notEmpty()) implies
(self.age > 18)
```

Άσκηση 2 (3/4)

- marryWife(p:Person)

```
context Person::marryWife(p:Person)
pre:    (self.sex == "male" or self.sex == nil) and (p.sex == "female" or
        p.sex == nil) and self.wife->isEmpty() and p.husband->isEmpty()
post:   self.wife == p and p.husband == self and self.sex == "male" and
        p.sex == "female"
```

- marryHusb(p:Person)

```
context Person::marryHusb(p:Person)
pre:    (self.sex == "female" or self.sex == nil) and (p.sex == "male"
        or p.sex=nil) and p.wife->isEmpty() and self.husband->isEmpty()
post:   p.wife == self and self.husband == p and self.sex == "female"
        and p.sex == "male"
```

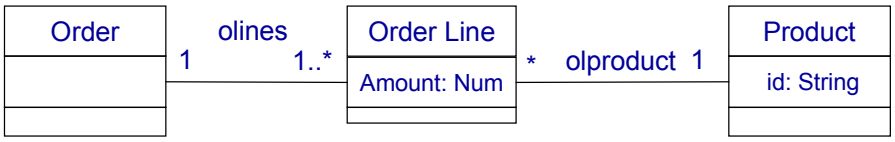

Άσκηση 2 (4/4)

- Divorce()

```
context Person::divorce()
pre:    self.wife->notEmpty() or self.husband->notEmpty()
post:   (self.sex == "male" implies (self.wife->isEmpty() and
        self.wife@pre.husband->isEmpty()))
        and
        (self.sex == "female" implies (self.husband->isEmpty() and
        self.husband@pre.wife->isEmpty()))
```

Άσκηση 3 (1/1)

Μια παραγγελία δεν πρέπει να έχει δύο ή περισσότερες παραγγελιογραμμές που να αναφέρουν το ίδιο προϊόν. Εκφράστε αυτόν τον περιορισμό σε OCL θεωρώντας το διάγραμμα της εικόνας.

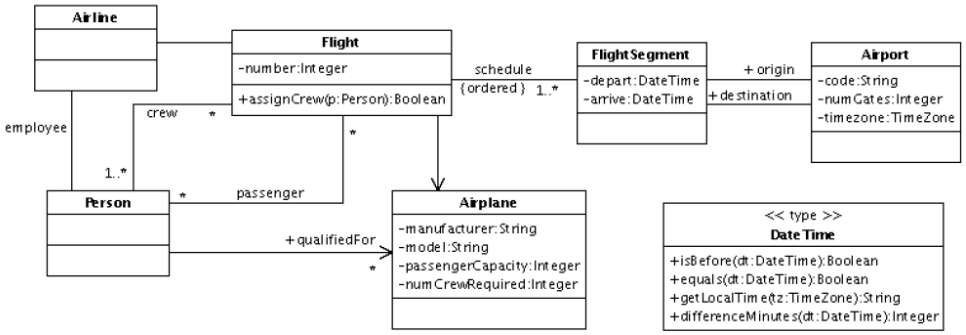


Λύση

```
context Order
inv:    self.olines->forAll(orderLine1, orderLine2 : OrderLine
        | (orderLine1 <> orderLine2) implies
        (orderLine1.olproduct.id <> orderLine2.olproduct.id))
```


Άσκηση 4 (1/3)

Θεωρώντας το διάγραμμα κλάσεων της εικόνας (για ένα σύστημα διαχείρισης αερογραμμών), εκφράστε τα ακόλουθα σε OCL:



Άσκηση 4 (2/3)

1. Ο αριθμός των επιβατών σε μία πτήση δεν πρέπει ποτέ να υπερβαίνει την χωρητικότητα σε επιβάτες του αεροπλάνου.

```
context Flight
inv: self.passenger->size() <= self.airplane.passengerCapacity
```

2. Όλα τα μέλη του πληρώματος μιας πτήσης πρέπει να είναι εργαζόμενοι της εταιρείας στην οποία ανήκει η πτήση.

```
context Flight
inv: crew->forAll(c:Person | c.employee == self.airline)
```

3. Το πλήθος του πληρώματος σε μία πτήση πρέπει να είναι ίσο με το ακέραιο γνώρισμα numCrewRequired για το αεροπλάνο.

```
context Flight
inv: self.crew->size() == self.airplane.numCrewRequired
```


Άσκηση 4 (3/3)

4. Ένα τμήμα (FlightSegment) μιας πτήσης δεν μπορεί να φτάσει πριν αναχωρήσει

```
context FlightSegment
inv: not arrive.isBefore(depart)
```

5. Το πρόγραμμα μίας πτήσης πρέπει να επιτρέπει τουλάχιστον 30 λεπτά στάση σε κάθε αεροδρόμιο

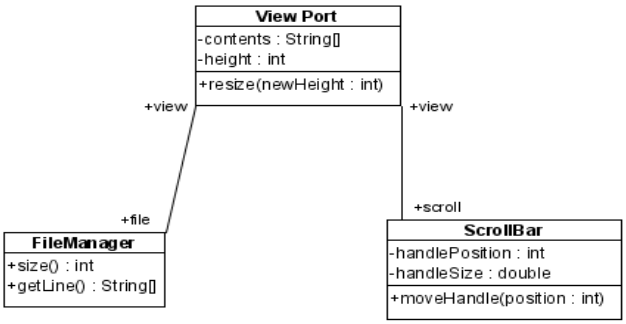
```
context Flight
inv: Sequence{ 1..schedule->size()-1 }->forAll(i |
     schedule->at(i+1).depart.differenceMinutes(schedule->at(i).arrive) >= 30)
```

6. Τις προυποθέσεις (pre-conditions) και τις μετασυνθήκες (post-conditions) της μεθόδου Flight::assignCrew

```
context Flight::assignCrew(p:Person):Boolean
pre: p.employee == self.airline
pre: p.qualifiedFor->includes(self.airplane)
post: result = self.crew->includes(p)
```

Άσκηση 5 (1/4)

Ο *Text Browser* είναι ένα εργαλείο για την εξερεύνηση ενός εγγράφου με κείμενο. Θεωρούμε πως ο Text Browser αποτελείται από τρία τμήματα, το *File Manager*, το *ViewPort* και το *ScrollBar*. Στην παρακάτω εικόνα έχουμε ένα UML class διάγραμμα που περιγράφει την εξωτερικά ορατή συμπεριφορά του Text Browser:



Άσκηση 5 (2/4)

•Ο File Manager είναι υπεύθυνος στο να παρέχει το κείμενο που θα δείξει ο ViewPort. Έχει δύο δημόσιες μεθόδους, την *getLine* και τη *size*. Θεωρούμε ότι το αρχείο αποτελείται μόνο από χαρακτήρες, χωρίζεται σε γραμμές και ότι μόνο ένα αρχείο μπορεί να εμφανιστεί τη φορά. Η getLine λαμβάνει ένα ακέραιο ως όρισμα και επιστρέφει τη γραμμή του αρχείου, με τους δείκτες να ξεκινάνε από τη γραμμή 0. Η size επιστρέφει ένα ακέραιο που δηλώνει τον αριθμό των γραμμών του αρχείου.

•Το ViewPort είναι υπεύθυνο για την εμφάνιση των γραμμών του κειμένου. Έχει δύο γνωρίσματα. Το πρώτο είναι ο ακέραιος *height* που δηλώνει τον αριθμό των γραμμών του κειμένου που αυτή τη στιγμή εμφανίζονται στο ViewPort. Το δεύτερο γνώρισμα είναι το *contents*, που είναι μια σειρά από γραμμές που αυτή τη στιγμή εμφανίζονται. Το ViewPort έχει και μία μέθοδο, τη *resize*. Η resize παίρνει ένα ακέραιο ως όρισμα και αλλάζει το μέγεθος του ViewPort έτσι ώστε να μπορεί να εμφανίσει τόσες γραμμές κειμένου.

•Το ScrollBar είναι υπεύθυνο για τον έλεγχο των γραμμών του αρχείου που αυτή τη στιγμή εμφανίζονται. Το ScrollBar έχει δύο γνωρίσματα και μία μέθοδο. Το πρώτο γνώρισμα ονομάζεται handlePosition και είναι ένας ακέραιος, οι τιμές του οποίου δείχνουν ποια γραμμή του αρχείου αυτή τη στιγμή εμφανίζεται στην κορυφή του ViewPort. Επομένως, οι τιμές του είναι μεταξύ του μηδενός και ενός λιγότερου του αριθμού γραμμών του αρχείου. Το δεύτερο γνώρισμα είναι το *handleSize* και είναι ένας πραγματικός αριθμός, μικρότερος ή ίσος του ένα, που δηλώνει το ποσοστό του αρχείου που αυτή τη στιγμή εμφανίζεται στο ViewPort. Η μέθοδος του ScrollBar ονομάζεται *moveHandle*. Λαμβάνει ένα μόνο ακέραιο όρισμα και μοντελοποιεί την κατάσταση στην οποία ο χρήστης έχει μετακινήσει το ScrollBar για να δει ένα διαφορετικό μέρος του αρχείου.

Άσκηση 5 (3/4)

Για να μοντελοποιηθούν πλήρως οι εργασίες ενός πραγματικού Text Browser, πρέπει να ισχύουν οι εξής πέντε ιδιότητες :

1) Όταν αλλάζει το μέγεθος του ViewPort, η νέα τιμή του height είναι ίδια με τη τιμή του ορίσματος της μεθόδου resize.

```
context ViewPort ::resize(newHeight:Integer):void
pre:        0 <= newHeight and newHeight <= self.maxSize
post:       self.height == newHeight
```

2) Όταν το ScrollBar μετακινείται, η νέα τιμή του handlePosition είναι ίδια με τη τιμή του ορίσματος της μεθόδου moveHandle.

```
context ScrollBar :: moveHandle (p:Integer) : int
pre:        0 <= p and p <= (view.file.size() - 1)
post:       self.handlePosition = p
```

3) Οι γραμμές που εμφανίζονται στο ViewPort που όλες προέρχονται από το αρχείο κειμένου, και το ViewPort εμφανίζει όσες γραμμές από το αρχείο μπορεί.

```
context ViewPort:
inv: self.contents->forAll(i:int|(0 <= i < self.height) and
(i < file->size() - scroll.handlePosition) implies
self.contents[i] = file->getLine(scroll.handlePosition + i))
```


Άσκηση 5 (4/4)

4) Η τιμή του handlePosition στο ScrollBar αντιστοιχεί στον αριθμό της γραμμής που εμφανίζεται στην κορυφή του ViewPort.

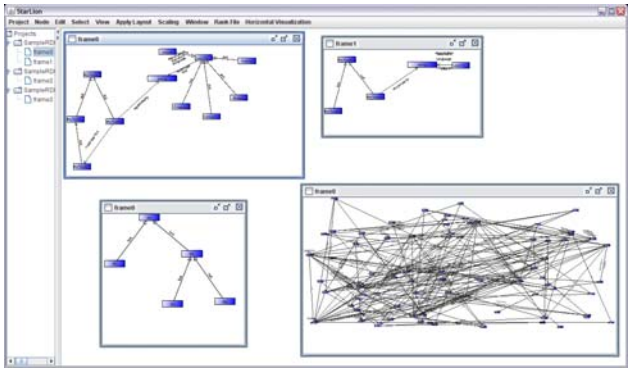
```
context ViewPort
inv: self.contents[0] == file.getLine(scroll.handlePosition)
```

5) Η τιμή του handleSize αντανακλά το ποσοστό των γραμμών από το αρχείο που εμφανίζονται στο ViewPort.

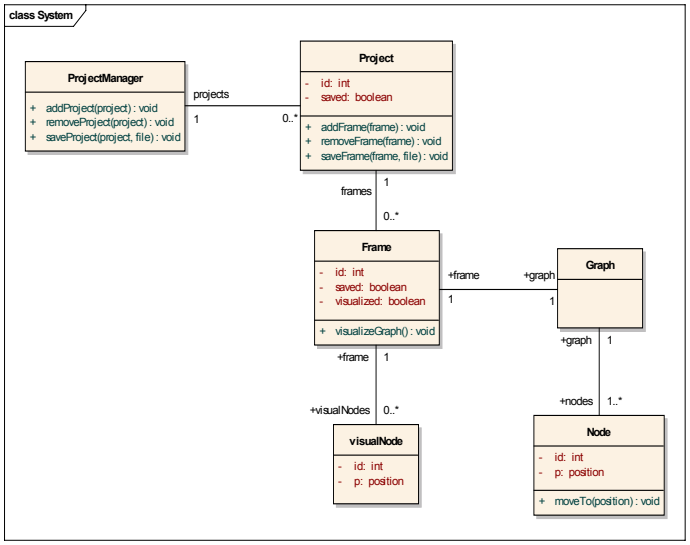
```
context ScrollBar
inv:
if view.file->size() <= view.height then self.handleSize == 1
else self.handleSize == view.height / view.file.size()
endif
```

Άσκηση 6 (1/5)

- Παρακάτω δίνεται το διάγραμμα κλάσεων ενός Συστήματος Οπτικοποίησης Γράφων. Το σύστημα αποτελείται από τον Project Manager ο οποίος είναι υπεύθυνος για την διαχείριση των Projects (άνοιγμα, κλείσιμο, αποθήκευση). Κάθε Project μπορεί να περιέχει πολλαπλά παράθυρα (Frames) και το καθένα από αυτό αναπαριστά έναν και μόνο ένα γράφο.



Άσκηση 6 (2/5)



Άσκηση 6 (3/5)

1. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου ProjectManager::addProject(project)

```
context ProjectManager::addProject(Project : project):void
pre: project <> Nil and not (self->projects->includes(project))
post: self->projects->size()@pre == (self->projects->size() + 1) and
      self->projects->includes(project)
```

2. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου ProjectManager::removeProject(project)

```
context ProjectManager::removeProject(Project : project):void
pre: project <> Nil and self->projects->size() > 0 and
     self->projects->includes(project)
post: self->projects->size()@pre == (self->projects->size() - 1) and
      not self->projects->includes(project)
```

3. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου ProjectManager::saveProject(project, file)

```
context ProjectManager::saveProject(Project : project, File : file):void
pre: project <> Nil and file <> Nil and self->projects->contains(project)
post: self->projects[project.id]->saved == true
```


Άσκηση 6 (4/5)

4. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου Project::addFrame(frame)

```
context Project::addFrame(Frame : frame):void
pre: frame <> Nil and not (self->frames->includes(frame))
post: self->frames->size()@pre == (self->frames->size() + 1) and
      self->frames->includes(frame)
```

5. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου Project::removeFrame(frame)

```
context Project::removeFrame(Frame : frame):void
pre: frame <> Nil and self->frames->size() > 0 and
     self->frames->includes(frame)
post: self->frames->size()@pre == (self->frames->size() - 1) and
      not self->frames->includes(frame)
```

6. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου Project::saveFrame(frame, file)

```
context Project::saveFrame(Frame : frame, File : file):void
pre: frame <> Nil and file <> Nil and self->frames->contains(frame)
post: self->frames[frame.id]->saved == true
```

Άσκηση 6 (5/5)

7. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου Frame::visualizeGraph()

```
context Frame::visualizeGraph():void
pre:  if(self.visualized == true) then (self->graph <> Nil and
      self->visualized == true and self->virtualNodes->size() == 0)
      else (self->graph <> Nil)
      endif
post: self->virtualNodes->size() == self->graph->nodes->size()
```

8. Καθορίστε τα invariants, pre-conditions και τα post-conditions της μεθόδου Node::moveTo(position)

```
context Node::moveTo(Position : position):void
pre:  position <> Nil
post: self->position == self->graph->frame->virtualNodes[self.id]
      .position
```


Χρήσιμες Αναφορές

- http://www.parlezuml.com/tutorials/umlforjava/java_ocl.pdf
- http://www.eclipse.org/articles/Article-EMF-Codegen-with-OCL/article.html