

Homework IV

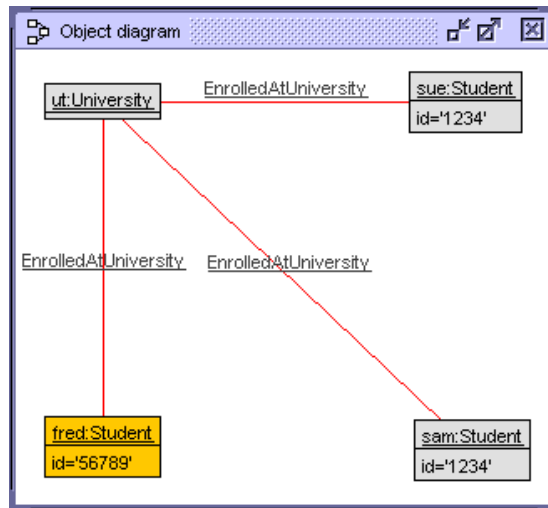
Gregory Williams
GW4975

EE 382C Requirements Engineering

12/03/2015

4.1.1

The invariant is violated by Fred who has an id with length equal to 5. The invariant `StudentIdMustBeLength4` works within the `University` context; it loads the set of all of the students that belong to `EnrolledAtUniversity` (i.e. are enrolled at the University) and checks each student's id to see if the length is equal to 4.



4.1.2

```
1 -- OCL constraints
constraints
3 context University
  -- A student's id number must be exactly
  -- four characters long
5   inv StudentIdMustBeLength4:
7     self.students->forAll( s | s.id.size() = 4 )
9   -- A student's id number must be unique
11  inv StudentIdMustBeUnique:
    self.students->forAll( s1, s2 | s1.id = s2.id implies s1 = s2 )
```

4.1.3

```
1 checking invariant (2) `University::StudentIdMustBeUnique`: FAILED.
  -> false : Boolean
3 Results of subexpressions:
   University.allInstances : Set(University) = Set{@ut}
5   self : University = @ut
   self.students : Set(Student) = Set{@fred,@sam,@sue}
7   s1 : Student = @fred
   s1.id : String = '56789'
9   s2 : Student = @fred
   s2.id : String = '56789'
11  (s1.id = s2.id) : Boolean = true
   s1 : Student = @fred
13  s2 : Student = @fred
   (s1 = s2) : Boolean = true
15  ((s1.id = s2.id) implies (s1 = s2)) : Boolean = true
   s1 : Student = @fred
17  s1.id : String = '56789'
   s2 : Student = @sam
19  s2.id : String = '1234'
   (s1.id = s2.id) : Boolean = false
21  ((s1.id = s2.id) implies (s1 = s2)) : Boolean = true
   s1 : Student = @fred
23  s1.id : String = '56789'
   s2 : Student = @sue
25  s2.id : String = '1234'
   (s1.id = s2.id) : Boolean = false
27  ((s1.id = s2.id) implies (s1 = s2)) : Boolean = true
   s1 : Student = @sam
29  s1.id : String = '1234'
   s2 : Student = @fred
31  s2.id : String = '56789'
   (s1.id = s2.id) : Boolean = false
33  ((s1.id = s2.id) implies (s1 = s2)) : Boolean = true
   s1 : Student = @sam
35  s1.id : String = '1234'
   s2 : Student = @sam
37  s2.id : String = '1234'
   (s1.id = s2.id) : Boolean = true
39  s1 : Student = @sam
   s2 : Student = @sam
41  (s1 = s2) : Boolean = true
   ((s1.id = s2.id) implies (s1 = s2)) : Boolean = true
43  s1 : Student = @sam
   s1.id : String = '1234'
45  s2 : Student = @sue
   s2.id : String = '1234'
47  (s1.id = s2.id) : Boolean = true
   s1 : Student = @sam
49  s2 : Student = @sue
   (s1 = s2) : Boolean = false
51  ((s1.id = s2.id) implies (s1 = s2)) : Boolean = false
   self.students->forAll(s1, s2 : Student | ((s1.id = s2.id) implies (s1 = s2))) : Boolean = false
53  University.allInstances->forAll(self : University | self.students->forAll(s1, s2 : Student |
   ((s1.id = s2.id) implies (s1 = s2)))) : Boolean = false
```

4.1.4

```

1  -- OCL constraints
constraints
3
context University
5    -- A student may be a GraduateStudent
    -- or an UndergraduateStudent
7    -- but not both
    inv StudentIsGradOrUndergradNotBoth:
9        self.undergraduates->intersection(self.graduates)->isEmpty()

```

4.1.5

```

1  checking invariant (1) `University::StudentIsGradOrUndergradNotBoth': FAILED.
-> false : Boolean
3  Results of subexpressions:
    University.allInstances : Set(University) = Set{@ut}
5    self : University = @ut
    self.undergraduates : Set(Student) = Set{@sam}
7    self : University = @ut
    self.graduates : Set(Student) = Set{@sam}
9    self.undergraduates->intersection(self.graduates) : Set(Student) = Set{@sam}
    self.undergraduates->intersection(self.graduates)->isEmpty : Boolean = false
11   University.allInstances->forall(self : University |
        self.undergraduates->intersection(self.graduates)->isEmpty) : Boolean = false

```

4.1.6

```

1  -- OCL constraints
constraints
3
context University
5    -- A student may not exceed the maxApprovedSemesterHours
    -- All Courses offered are assumed to have 3 credit hours
7    inv StudentIsGradOrUndergradNotBoth:
        self.students->forall(s | s.takingCourses->size() * 3 <= s.maxApprovedSemesterHours)

```

4.1.7

```

1  checking invariant (1) `University::StudentIsGradOrUndergradNotBoth': FAILED.
2  -> false : Boolean
3  Results of subexpressions:
4    University.allInstances : Set(University) = Set{@ut}
    self : University = @ut
6    self.students : Set(Student) = Set{@sam}
    s : Student = @sam
8    s.takingCourses : Set(Course) = Set{@BUS311,@CS306,@E306,@EE302,@EE323,@EE338,@EE379K}
    s.takingCourses->size : Integer = 7
10   3 : Integer = 3
    (s.takingCourses->size * 3) : Integer = 21
12   s : Student = @sam
    s.maxApprovedSemesterHours : Integer = 18
14   ((s.takingCourses->size * 3) <= s.maxApprovedSemesterHours) : Boolean = false

```

```

self.students->forall(s : Student | ((s.takingCourses->size * 3) <= s.maxApprovedSemesterHours)) :
  Boolean = false
16 University.allInstances->forall(self : University | self.students->forall(s : Student |
  ((s.takingCourses->size * 3) <= s.maxApprovedSemesterHours))) : Boolean = false

```

4.2.1

```

-- OCL constraints
2 constraints

4 context Student
  inv studentEnrolledInUniversity: self.isEnrolledAt.students->includes(self)

6
8 context Student :: drop(c : Course)
  pre studentIsRegistered: self.takingCourses->includes(c)
  pre studentHasMoreThanOneClass: self.takingCourses->size() > 1
10 post studentIsNotRegistered: not self.takingCourses->includes(c)
  post studentDidNotDropOtherCoursesRegistered: self.takingCourses->including(c) =
    self.takingCourses@pre
12 post droppedCourseNotFull: c.isFull = false
  post studentStillEnrolledInUniversity: self.isEnrolledAt = self.isEnrolledAt@pre
14 post onlyThisStudentWasRemoved: c.studentsEnrolled->including(self) = c.studentsEnrolled@pre

```

4.2.2

```

!create ut : University
2 !create sam : Student
!create sue : Student
4 !insert (sam,ut) into EnrolledAtUniversity
!insert (sue,ut) into EnrolledAtUniversity
6 !create EE302 : Course
!create CS306 : Course
8 !create BUS311 : Course
!create EE411 : Course
10 !create EE379K : Course
!create E306 : Course
12 !create EE338 : Course
!create EE323 : Course
14 !insert (sam,EE302) into TakingCourse
!insert (sam,CS306) into TakingCourse
16 !insert (sam,BUS311) into TakingCourse
!insert (sam,EE323) into TakingCourse
18 !insert (sam,EE379K) into TakingCourse
!insert (sam,E306) into TakingCourse
20 !insert (sam,EE338) into TakingCourse
!insert (sue,EE302) into TakingCourse
22 !insert (sue,CS306) into TakingCourse
!insert (sue,BUS311) into TakingCourse
24 !insert (sue,EE323) into TakingCourse
!set EE302.isFull := true
26 !openter sam drop(EE302)
!delete (sam,EE302) from TakingCourse
28 !set EE302.isFull := false
!opexit

```

4.2.3

```
1 CR2.cmd> !openter sam drop(EE302)
precondition `studentIsRegistered' is true
3 precondition `studentHasMoreThanOneClass' is true
CR2.cmd> !delete (sam,EE302) from TakingCourse
5 CR2.cmd> !set EE302.isFull := false
CR2.cmd> !opexit
7 postcondition `studentIsNotRegistered' is true
postcondition `studentDidNotDropOtherCoursesRegistered' is true
9 postcondition `droppedCourseNotFull' is true
postcondition `studentEnrolledInUniversity' is true
11 postcondition `onlyThisStudentWasRemoved' is true
```