Abstract

This article proposes a system for generating possible *University Classes Schedules*. It uses multi-agent negotiation to find satisfactory solutions to the problem, while trying to consider *personal preferences* of the represented people and institutions.

1 University Classes

A class is an en event, that brings together a *group of students*, and a *professor* in certain *classroom* in order to learn/teach the specified *discipline*.

It happens periodically, usually weekly, at the established *day of week* and *time*.

```
data GroupRef     = GroupRef String
data ProfessorRef = ProfessorRef String
data ClassroomRef = ClassroomRef String
data Discipline = DisciplineClass { disciplineId :: String
                                  , disciplineMinutesPerWeek :: Int
                                  }
                | DisciplineLab   { disciplineId :: String
                                  , disciplineMinutesPerWeek :: Int
                                  }
  -- redefined 'System.Time.Day' - no 'Sunday'
data Day = Monday | Tuesday | Wednesday
         | Thursday | Friday | Saturday
deriving (Eq, Ord, Enum, Bounded, Ix, Read, Show)
data Class time = Class { classDay        :: Day
                        , classBegins     :: time
                        , classEnds       :: time
                        , classDiscipline :: Discipline
                        , classGroup      :: GroupRef
                        , classProfessor  :: ProfessorRef
                        , classRoom       :: ClassroomRef
                        }
```

The classes are negotiated by the interested parties: 1) students / groups, 2) professors, 3) classrooms.

Each negotiation participant has a *timetable*, holding a schedule for one week, that repeats throughout the academic period. The *timetable* is actually a table: the columns represent days of week; the rows – discrete time intervals. Actual timetable structure may vary, as can be seen in figure 1.

```
class (Ord t, Bounded t, Show t) ⇒ DiscreteTime t where
  toMinutes   :: t → Int
  fromMinutes :: Int → t
```

	Mon	Tue	Wed	Thu	Fri	Sat
08:30 – 09:00						
09:00 – 09:30						
09:30 – 10:00						
10:00 – 10:30						
10:30 – 11:00						
11:00 – 11:30						
11:30 – 12:00						
⋮ ⋮						

(a) Timetable without recesses.

	Mon	Tue	Wed	Thu	Fri	Sat
08:30 – 09:10						
09:15 – 09:55						
10:05 – 10:45						
10:50 – 11:30						
11:40 – 12:20						
12:25 – 13:05						
13:15 – 13:55						
⋮ ⋮						

(b) Timetable with recesses.

Figure 1: Possible *timetable* structures.

```
class (DiscreteTime time) ⇒ Timetable tt x time | tt → time
                                                  , x  → time
  where classesOn :: tt → Day  → [Class time]
        classesAt :: tt → time → [(Day, Maybe x)]
        classAt   :: tt → Day → time → Maybe x
```