K3DataModel

```
Model K3DataModel begin
  Library Standard;
```

```
  // Declaration of the k3 core classes
  abstract Class @Section                                         // r1, d7
                 "Training session for a class held at a specific date
                  (day and time)";
  Class          Lecture                   extends Section;      // d2
  Class          ClinicalLabSection        extends Section;

  abstract Class @°Class
                 "Set of sections constituting acomplete course on
                  a given topic";
  Class          NonClinicalClass          extends °Class;       // d1
  Class          ClinicalClass             extends °Class;       // d1

  Class          @ProgramOfStudy                                  // d4
                 "Set of classes required to obtain a degree";

  Class          @SequenceOfClasses                               // r1
                 "Set of classes, all being taught during the same
                  specified quarter. Can also include sections not
                  belonging to any class";

  abstract Class @Person                                          // d9
                 "Information in the system about a person
                  relevant to the system";
  Class          Student                   extends Person;       // d9
  Class          Instructor                extends Person;       // d9
  Class          StaffMember               extends Person;       // d9
  Class          ClinicalSiteAdministrator extends Person;       // d9

  Class          @Cohort                                          // r18
                 "Set of all students registered to the same Nursing
                  program of study, for the same quarter"
                  extends Student {};

  Class          @Department "or Department Section";

  Class          @ClinicalSite;

  external abstract Class
                 #User "of the system";
  Class          PA  "Program Administrator" extends User;
  Class          NSM "Nursing Staff Member"  extends User;
```

```
// Declaration of auxiliary classes
Class          @Information "provided in a report"      is deferred;
Class          @RoomType                                is deferred;
constant Class @Name "of an item managed by the system" is deferred;
constant Class @Id   "of an item managed by the system" is deferred;
Class          @Credit "of a student"                   is deferred;
Class          @GPA "of a student" /* whatever it is */ is deferred;
constant Class @Need "of an item managed by the system" is deferred;
Class          @Date
               "From which one can derive year,
                month, day of month and time of day"    is deferred;
Class          Month   extends Date
               "First day of a month,   at 00:00"       is deferred;
Class          Quarter extends Date
               "First day of a quarter, at 00:00"       is deferred;

Class          @EMail                                   is deferred;
Class          @PhoneNumber                             is deferred;
Class          @Address                                 is deferred;

Class          @EReport "issued by the system on request"
               extends Event;

Class          @ENotification "issued by the system to warn users"
               extends Event;
```

```
// Declaration of ancillary functions
Duration quarterDuration (Quarter quarter) is deferred;
Boolean  overlap (Date date1, Duration d1, Date date2, Duration d2)
         "Whether the two time intervals overlap"
         is deferred;
Boolean  inQuarter (Date date, Quarter quarter) is
         overlap (date, 1*s, quarter, quarterDuration (quarter));
```

```
// Declaration of the k3 core objects
main Object #system
  "A University-wide information management system"
begin
  Department            nursingDepartment
                        "One of the Departments";
  ProgramOfStudy {}     programs
                        "of study of all Departments";
  ProgramOfStudy {}     nursingPrograms
                        "of study of the Nursing Department";
  Cohort {}             cohorts
                        "managed by the system";
  Cohort {}             nursingCohorts
                        "following a program of study of the
                         Nursing Department"
    is all x of cohorts
       such that x.programOfStudy in nursingPrograms;
  ClinicalSite {}       clinicalSites;
  °Class {}             classes;
  ClinicalClass {}      clinicalClasses :
    Guarantee consistency is ensure me in classes;
  SequenceOfClasses {}  sequencesOfClasses;
  Section {}            sections;
  ClinicalLabSection {} clinicalLabSections :
    Guarantee consistency is ensure me in sections;
  Person {}             persons "relevant to the system"; // d9
  Student {}            students;                         // d9
  Student {}            nursingStudents is
    all x of students such that x.cohort in nursingCohorts;
end system;
```

```
end K3DataModel;
```

```
// Definition of the k3 core classes
°Class begin
  Name       name;                                     // d2, d3
  Id         id;                                       // r17
  Need {}    lectureRoomNeeds;                         // d2, d3
  Section {} sections is
    all x of system.sections such that x.class = me;
  Section {} sectionsInQuarter (Quarter quarter) is
    all x of sections such that inQuarter (x.beginning, quarter);
  Student {} students is
    all x of system.students such that me in x.classes;
end °Class;
```

```
NonClinicalClass begin
  Need {} instructorNeeds;                             // d2
end NonClinicalClass;
```

```
ClinicalClass begin
  Need {} clinicalSiteNeeds;                           // d3
  Need {} lectureInstructorNeeds;                      // d3
  Need {} clinicalLabInstructorNeeds;                  // d3
end ClinicalClass;
```

```
Section begin
  °Class      class      "to which the section belongs";
  Date        beginning  "when";              // d7
  Duration    @°duration "for how long";      // d7
  Student {}  students  "registered to the section";
  Information contactInformation;
end Section;
```

```
ClinicalLabSection begin
  Name         name;
  Instructor   instructor;  // d7
  ClinicalSite site;
  Department   department;
end ClinicalLabSection;
```

```
ProgramOfStudy begin
  Name       name;                // d4
  °Class {} requiredClasses;      // d4
end ProgramOfStudy;
```

```
Class @ClassQuarter begin
  °Class  class;
  Quarter quarter;
end ClassQuarter;
```

```
SequenceOfClasses begin
  ClassQuarter {} classesQuarters;
  °Class {}       classes  is classesQuarters.class;
  Quarter {}      quarters is classesQuarters.quarter;
  Section {}      addedSections
                  "not in any of the classes of classesQuarters";
  Section {}      sections is
    UNION (for all x of classesQuarters : x.class.sectionsInQuarter(x.quarter))
    union     addedSections;
end SequenceOfClasses;
```

```
Cohort begin
  value                is all x of Student such that x.cohort = me;
  Name                 name;
  Id                   id;
  ProgramOfStudy       programOfStudy;
  Month                startMonth;
  SequenceOfClasses    preferredSequence;
  Guard consistency1 is
    ensure preferredSequence.classes = programOfStudy.requiredClasses;
  Guard consistency2 is
    ensure AND (preferredSequence.quarters >= startMonth);
end Cohort;
```

```
Person begin
  Name        name;
  Id          id;
  EMail       eMail;         // d9
  PhoneNumber phoneNumber;   // d9
end Person;
```

```
Student begin
  Date                     admissionDate;
  Boolean                  @partTime: default is false;
  Cohort                   cohort "in which the student is registered";
  SequenceOfClasses        sequenceOfClasses :
                           default is cohort.preferredSequence;
  °Class {}                classes "to which the student is registered"
                           is sequenceOfClasses.sections.class;
  ClinicalClass {}         clinicalClasses
                           "to which the student is registered"
                           is all x of classes
                              such that x in ClinicalClass;
  NonClinicalClass {}      nonClinicalClasses
                           "to which the student is registered"
                           is all x of classes
                              such that x in NonClinicalClass;
  Section {}               sections
                           "to which the student is registered"
                           is all x of classes.sections
                              such that me in x.students;
  ClinicalLabSection {}    clinicalLabSections
                           "to which the student is registered"
                           is all x of sections
                              such that x in ClinicalLabSection;
  Lecture {}               lectures
                           "to which the student is registered"
                           is all x of sections
                              such that x in Lecture;
end Student;
```

```
Department begin
  °Class {} classes "Classes offered by this Department";
end Department;
```

```
ClinicalSite begin
  Name        name;                 // d8
  Person      contactPerson;        // d8
  Information contactInformation;   // d8
  Address     address;
end ClinicalSite;
```

```
EReport begin
  occurrence is specific;
  Information {} contents is specific;
end EReport;
```

```
ENotification begin
  occurrence is specific;
  Person {} persons "to be notified" is specific;
end ENotification;
```