# Ontology Building

Present by :
Umi Laili Yuhana
[1] Computer Science & Information Engineering National Taiwan University
[2] Teknik Informatika Institut Teknologi Sepuluh Nopember ITS Surabaya Indonesia
11/04/2007

# Outline

- Reason
- Definition of Ontology
- Building Ontology
- Reference

# Reason

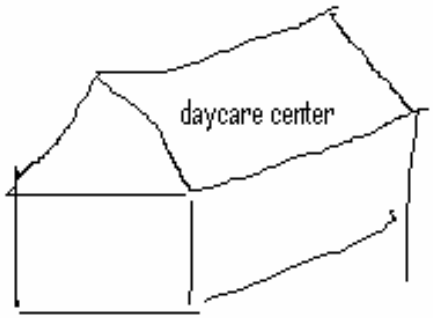
- To Share common understanding of the structure of information among people or software agents
- To enable reuse of domain knowledge
- To make domain assumptions explicit
- To separate domain knowledge from the operational knowledge
- To Analyze domain knowledge

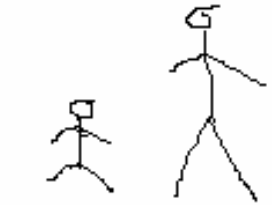
# What is an Ontology

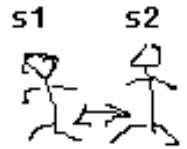
- Kind of things that actually exist, and how to describe them -> philosophy term
- In computer science :
  - Explicit and formal specification of a conceptualization
  - Consist of finite list of terms and the relationships between these terms

# Building Ontology

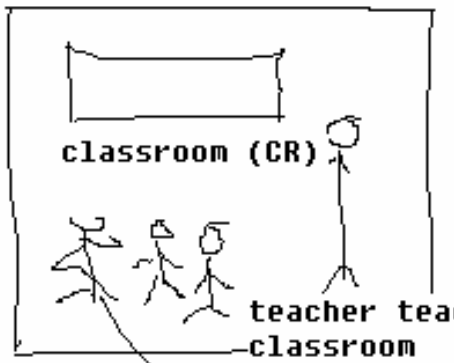
## Case study : Daycare Ontology

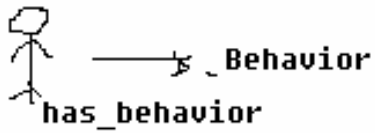


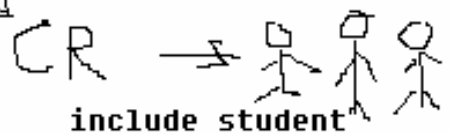




has_classmate




has_behavior




include_student

A small child care consisting :
- 3 classroom (2 in the morning, 1 in the afternoon)
- 3 teachers (each teacher assign to 1 classroom)
- handfull of children

Some students have behavior that can endanger person. Each student has teacher.

# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Determine Scope (Q)

- Basic questions :
  - What is the domain that the ontology will cover ?
  - For what we are going to use the ontology ?
  - For what types question should the ontology provide answer ?
  - Who will use and maintain the ontology ?

# Determine Scope (A)

- Answer :
  - What is the domain that the ontology will cover ? → Small child care center / daycare
  - For what we are going to use the ontology ?
    - Infer knowledge about student's negative behaviors to which s/he will be exposed
  - For what types question should the ontology provide answer ?
    - Who is the classmates of each student ?
    - What is negative behavior of each student ?
  - Who will use and maintain the ontology ?
    - Teachers, to care the student and avoid the student from negative behavior of her/his classmate

# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Consider Reuse

- We can reuse ontology in the same domain knowledge if it exist
- If no ontology exist, create new one

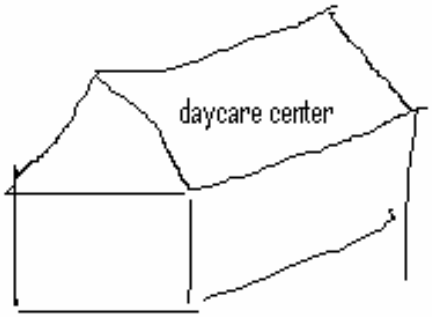
# Step by step Build Ontology

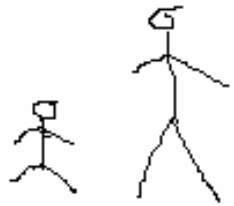
- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

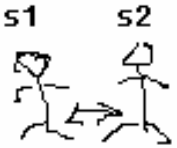
# Enumerate Terms

- Identify relevant terms
- Write down in an unstructured list all the relevant terms
- Noun -> basis for class names
- Verbs -> basis for property names
  - *Is part of*
  - *Has component*
  - *etc*

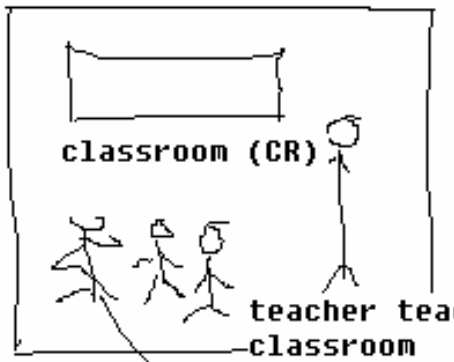
# Enumerate Terms (cont.)

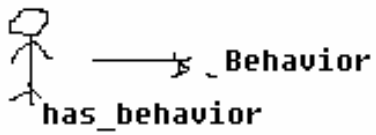


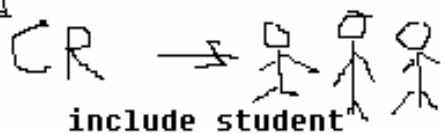




has_classmate




has_behavior




include_student

A small child care consisting :
- 3 classroom (2 in the morning, 1 in the afternoon)
- 3 teachers (each teacher assign to 1 classroom)
- handfull of children

Some students have behavior that can endanger person. Each student has teacher.

# Enumerate Term (cont.)

- Classroom
- Student
- Teacher
- Behavior
- Person
- behavior_of
- has_behavior
- is_practice_by
- endanger
- is_exposed_to
- teach
- is_taught
- includes_student
- attends_classroom
- has_teacher
- teach_student
- has_classmate
- has_date_of_birth
- has_age

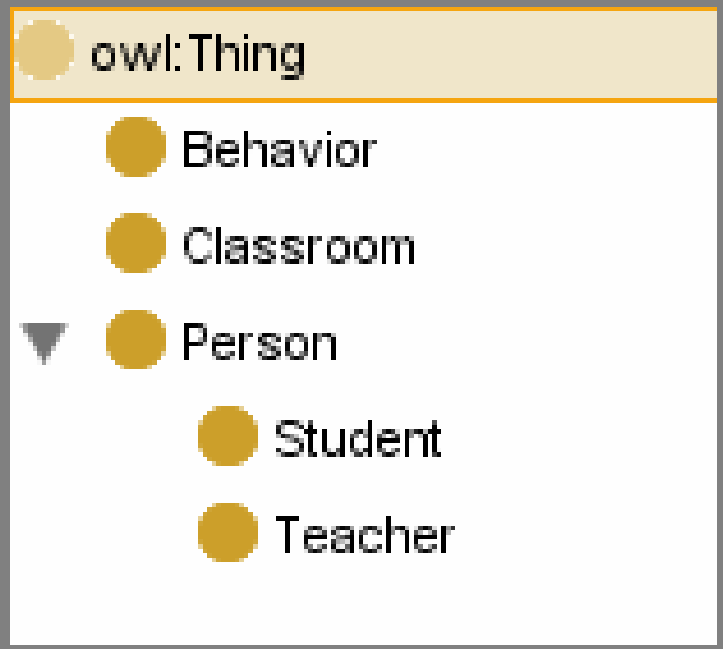
# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Define Taxonomy

- Organize relevance terms in taxonomic (subclass) hierarchy
- Terms as class : Classroom, Student, Teacher, Person, Behavior

# Define Taxonomy (cont.)



# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Property

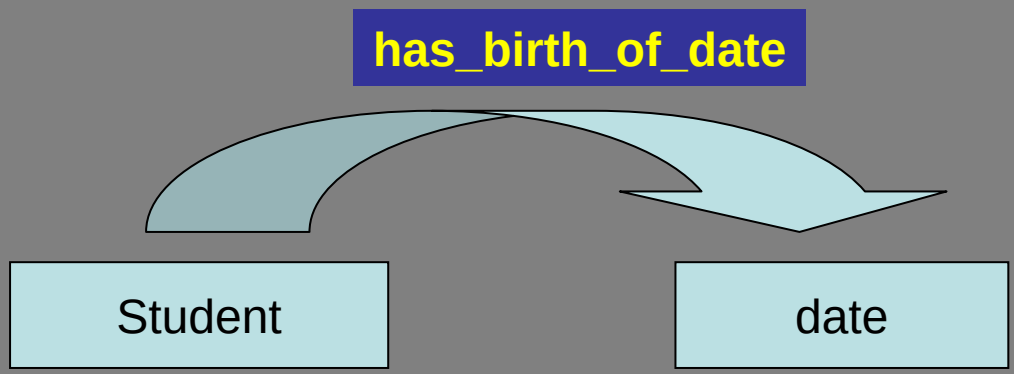
- Property : represent relationships between two individual
- Property = Slot
- Link individual from the domain and individual from the range
- There are 3 properties :
  - Object Properties
  - Data Type Properties
  - Annotation Properties

# Object Property

- Link an individual to an individual
- Types of object property
  - Inverse property
    - e.g. : has_parent inverse of has_child
  - Functional property
    - e.g. : has_birth_mother
  - Transitive property
    - e, g. : has_anchestor
  - Symmetric property
    - e. g. : has_sibling

# Data Type Property

- Link an individual to an XML schema data type value or an rdf literal
- e. g. :



# Annotation Property

- Used to add information (metadata – data about data) to classes, individuals and object / data type property

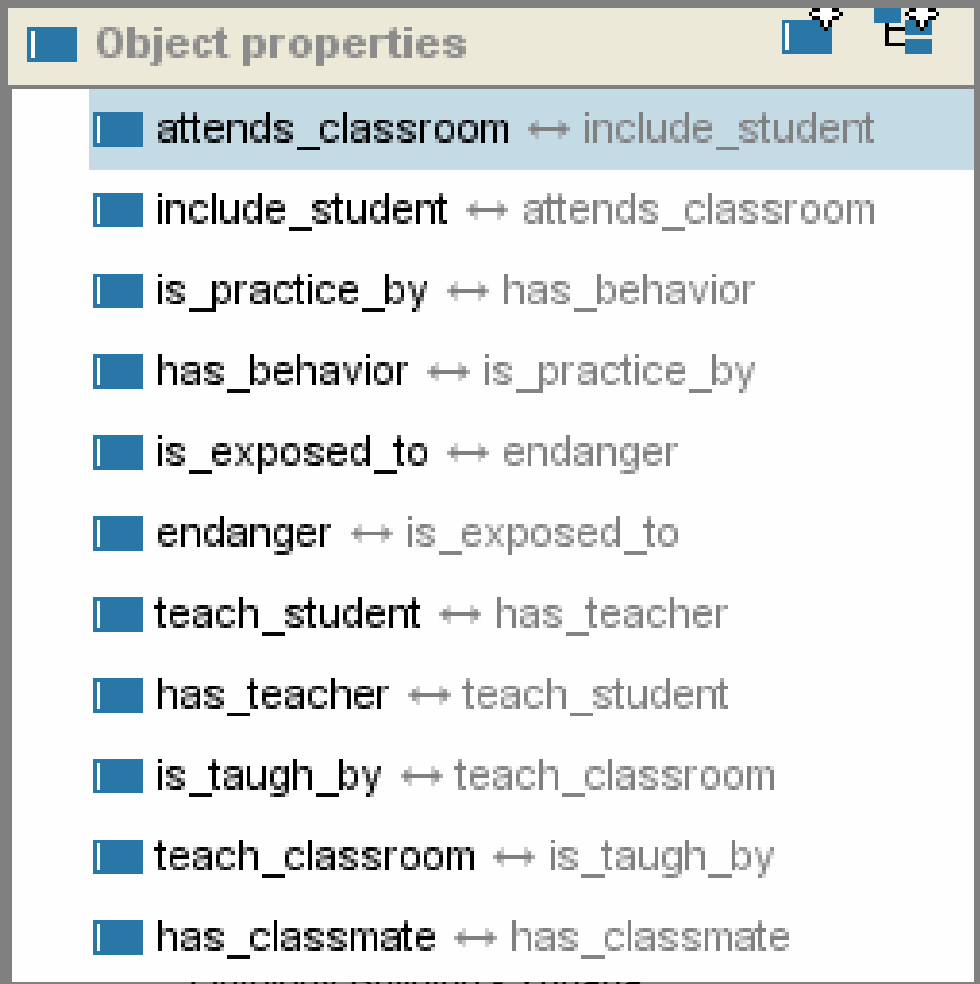
# Define Property

| No | Property | Domain | Range |
|---|---|---|---|
| 1 | has_behavior | Student | Behavior |
| 2 | is_practice_by | Behavior | Student |
| 3 | endanger | Behavior | Person |
| 4 | is_exposed_to | Person | Behavior |
| 5 | teach | Teacher | Classroom |
| 6 | is_taught | Classroom | Teacher |
| 7 | Includes_student | Classroom | Student |
| 8 | attends_classroom | Student | Classroom |
| 9 | has_teacher | Student | Teacher |
| 10 | teach_student | Teacher | Student |
| 11 | has_classmate | Student | Student |
| 12 | has_date_of_birth | Student | Date |
| 13 | has_age | Student | Int |

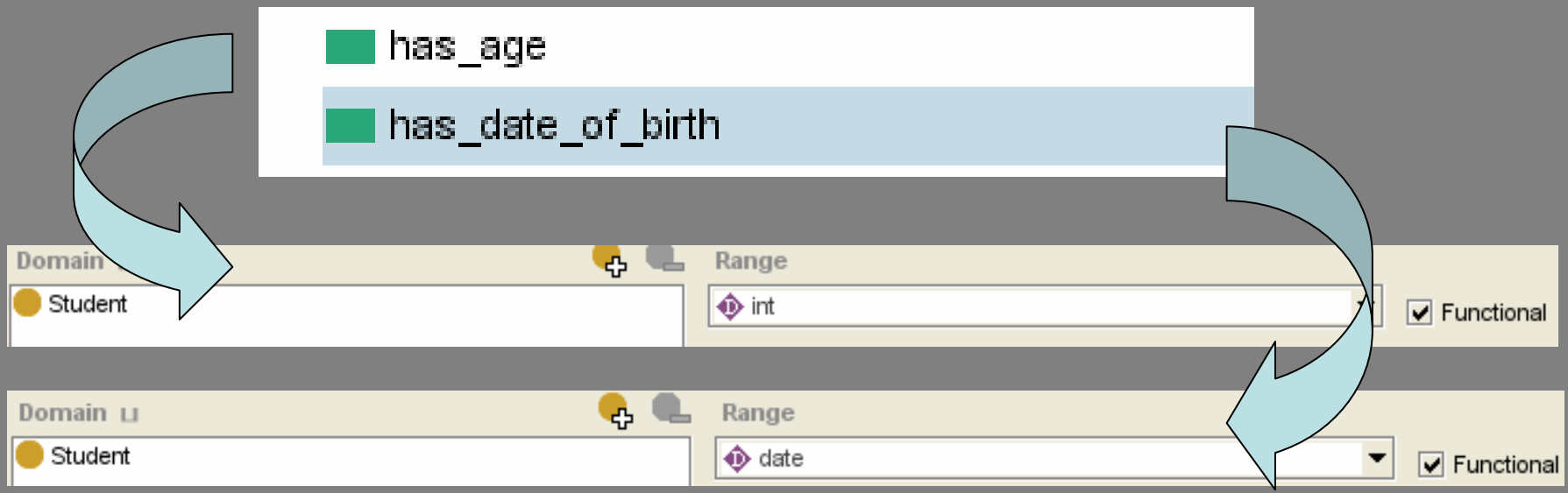
# Define Property (cont.)

- has_behavior is inverse of  is_practice_by
- endanger is inverse of is_exposed_to
- teach is inverse of is_taught
- includes_student is inverse of attends_classroom
- has_teacher is inverse of teach_student
- has_classmate is symmetric property
- has_date_of_birth is data type property
- has_age is data type property

# Object Properties



# Data Type Property



# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Facets

- Facet is used to represent information about properties (slots), sometimes called role restrictions
- Kind of facets :
  - Cardinality
  - Value Type

# Cardinality

- Cardinality represents the exact number of values that may be asserted for the slot for that class
  - Single cardinality
  - Multiple cardinality
    - Minimum cardinality
    - Maximum cardinality

# Value Type

- Value-type facet describes what type of values can fill in the slot
  - String
  - Number
  - Boolean
  - Enumerated

# Cardinality

- Cardinality represents the exact number of values that may be asserted for the slot for that class
- Cardinality
  - Minimum cardinality
  - Maximum cardinality
- Value Type

# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Instance

- Object or individual of class
- Example :
  - Instances of student :
    - Ariel,
      - has_date_of_birth : November 15, 2004
      - has_behavior : throwing_toys
      - Has_teacher : miss_Lyn

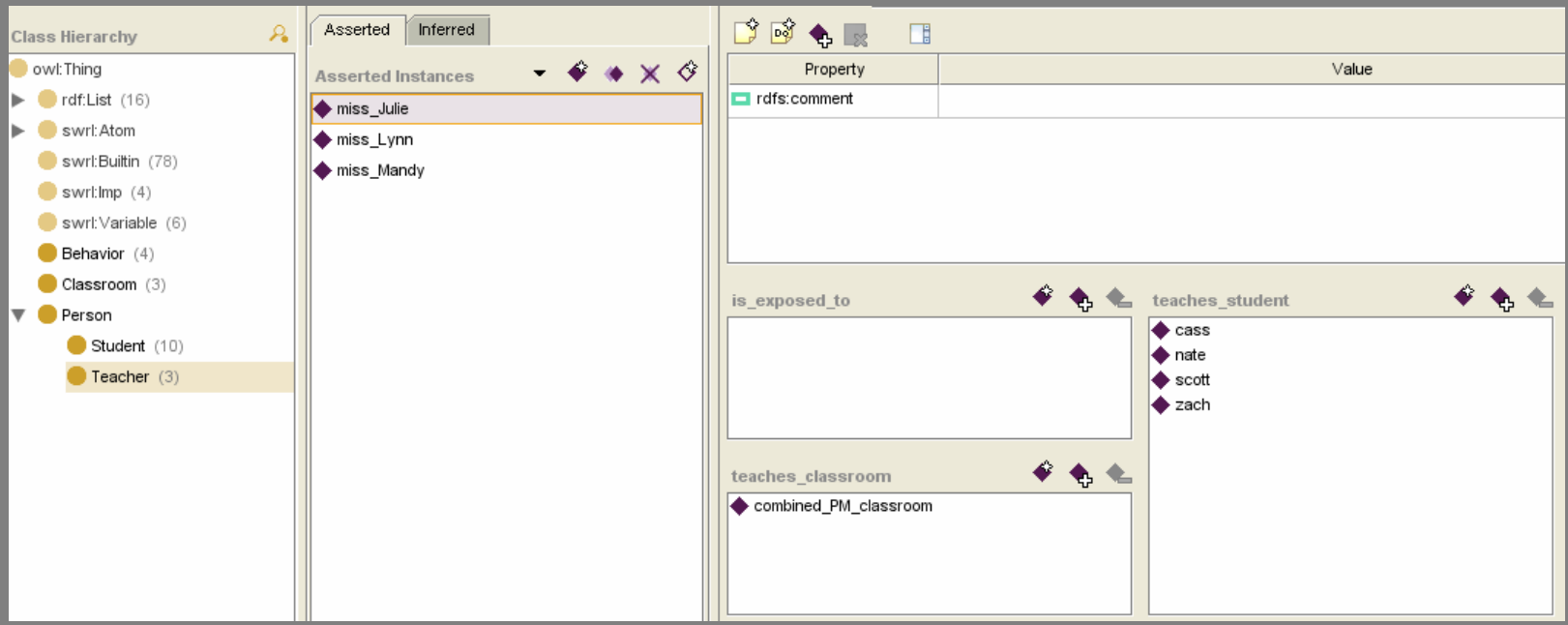
# Instances

| Instance of Class | is_taugh_by |
|---|---|
| combine_PM_classroom | miss_Julie |
| older_kids_AM_classroom | miss_Mandy |
| younger_kids_AM_classroom | miss_Lyn |

| Instance of Student | has_date_of_birth | has_behavior | has_teacher |
|---|---|---|---|
| ariel | November 15, 2004 | Throwing_toys | miss_Lyn |
| cal | September 12, 2003 | | miss_Mandy |
| cass | January 20, 2005 | Throwing_toys | miss_Julie |
| ella | June 15, 2004 | Bitting, Pinching | miss_Lyn |
| ginny | December 20, 2003 | Hitting | miss_Mandy |
| jeremy | April 24, 2003 | Throwing_toys | miss_Mandy |
| katie | March 14, 2005 | Bitting | miss_Lyn |
| nate | December 22, 2003 | | miss_Mandy, miss_Julie |
| scott | September 9, 2004 | Bitting | miss_Lyn, miss_July |
| zach | July 10, 2003 | | miss_Julie, miss_Mandy |

| Instance of Teacher | teaches_classroom | teach_students |
|---|---|---|
| miss_Julie | combine_PM_classroom | cass,nate,scot,zach |
| miss_Lyn | younger_kids_AM_classroom | ariel,ella,katie,scott |
| miss_Mandy | older_kids_AM_classroom | cal,ginny,jeremy,nate,zach |

# Fill instances in protégé



# Step by step Build Ontology

- Determine scope
- Consider reuse
- Enumerate Terms
- Define Taxonomy
- Define Properties
- Define Facets
- Define Instances
- Check for Anomalies

# Check Anomalies

- Check anomalies or consistency with reasoner
  - Pellet
  - Racer

# Conclusion

- There is no single correct ontology for any domain
- Quality of ontology can be proofed by using it in applications

# Reference


- N.F Noy, and D.L. McGuinnes, Ontology Development 101 : A Guide to Creating Your First Ontology, 2001
- Horridge, Mattahew, A Practical Guide to Building OWL_Ontologies Using The Protege-OWL plugin and CO-ODE Tool, The University of Manchester, 2004
- https://mywebspace.wisc.edu/jpthielman/web/DaycareOntology.htm