Carleton University
Systems and Computer Engineering
SYSC5708 Model-Driven Development of Real-Time and Distributed Software

ASSIGNMENT 2: Defining a UML Profile

POSTED: Wednesday, Oct. 22, 2014

DUE: Monday, Nov.10, in class or soft copy by email in a *single PDF file.*

The objective of this assignment is to design a UML profile for Entity-Relationship (ER) models used for database design. The project contains three steps:

a) Design the *domain model* describing the concepts necessary for building ER models;

b) Define a *UML profile* by mapping the domain concepts to UML;

c) Apply the profile to build an ER model for a given example.

The ER model is introduced first (see R. Elmasri, S. Navathe, "Fundamentals of Database Systems" or other database textbooks for more information).

Entity-Relationship data model

The entity-relationship (ER) data model views the real world as a set of basic objects (**entities**) and **relationships** among these objects. Entities and relationships can both have **attributes**. The ER model is a high-level conceptual data model used in the database design process. Usually, the graphical representation of ER models is in the form of ER diagrams, which uses the notation given in Fig.1. An example of ER diagram is given in Fig.2.

An ER diagram represents **entity types** as rectangles. An entity type is a set of entities with the same attributes (which are represented as ellipses connected to their owning entity sets by a line). An important property of an entity type is that it has a **key** represented by an attribute (single-valued or composite) that has a unique value for every element in the entity set. The key attribute is underlined in the diagram.

Attributes in an ER model may be further described as *single-valued*, *multi-valued*, *composite*, or *derived*:

- a single-valued attribute has a single value (e.g., the age of a person)
- a multi-valued attribute (illustrated with a double-line ellipse) may have more than one value for at least one instance of its entity (e.g., the degrees of a person).
- a composite attribute may itself contain two or more attributes and is indicated as having contributing attributes of its own (e.g., the address of a person).
- a derived attribute is one whose value is entirely determined by other information in the database; it is indicated by a dashed ellipse.

Entity types may be either *strong* or *weak*:

- a strong entity type (drawn as single line rectangle) is uniquely defined by its attributes alone (i.e., the key is represented by one of its own attributes)

- a weak entity type (drawn as a double line rectangle) isn't uniquely identified by its own attributes, meaning that two entities may have completely identical attribute values. The entities of a weak type are identified by being related to specific entities from another entity type named *identifying owner* through a so-called *identifying relationship* (drawn as a double-line diamond).

A **relationship type** R among n entity types E_1, E_2 ,..., E_n, is a set of associations among entities from these types. Relationships are drawn in a ER diagram as diamonds. Mathematically, a relationship type R is a set of relationship instances r_i, where each r_i is an n-tuple of entities $(e_1, e_2, ..., e_n)$, such that $e_1 \in E_1$,..., $e_n \in E_n$. Note that relationships among two entity types are named *binary* (i.e., $n=2$), among three entity types *ternary* (i.e., $n=3$), among three entity types *quaternary* (i.e., $n=4$), etc. The most frequently used are binary relationships.

Relationship types have certain constraints that limit the possible combinations of entities participating in relationship instances. Two such constraints are *cardinality ratio* and *participation*.

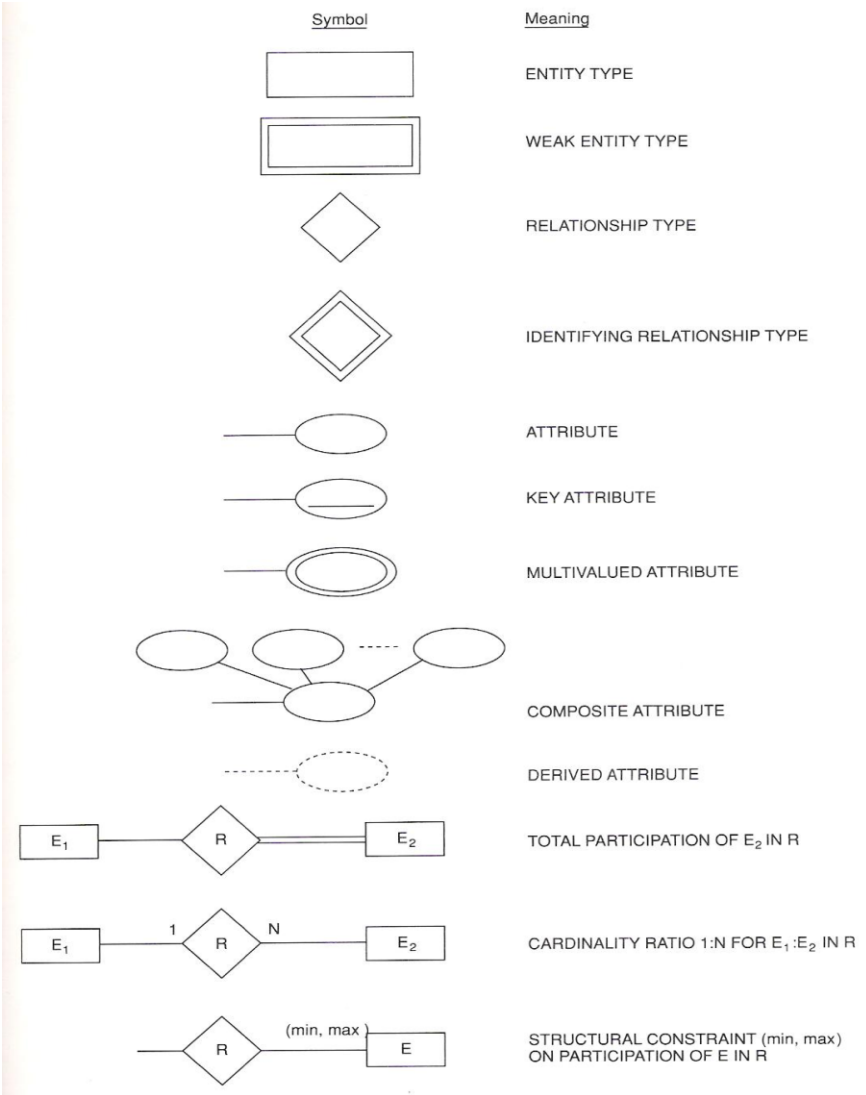

Figure 1. Graphical notation for ER diagrams

- the *cardinality ratio* specifies the number of relationship instances an entity can participate in; common cardinality ratios for binary relationships types are *1:1, 1:N, M:N*.
- the *participation constraint* (which may be *total* or *partial*) expresses whether all entities of an entity type participate in a given relationship or not.

The cardinality ratio and participation constraint can be expressed together as the *structural constraint* of a relationship type, given as a pair of integers *(min, max)*, where *0 <= min <= max* and *max >= 1*. For each entity $e \in E$, *e* must participate in at least *min* and at most *max* relationship instances $r \in R$ at all times. If *min=0*, this implies partial participation, whereas *min>0* implies total participation.

Example of ER diagram

In order to facilitate the understanding of the above ER notation, the ER diagram for a database that keeps track of a company's employees, departments, and projects is given in Fig.2.

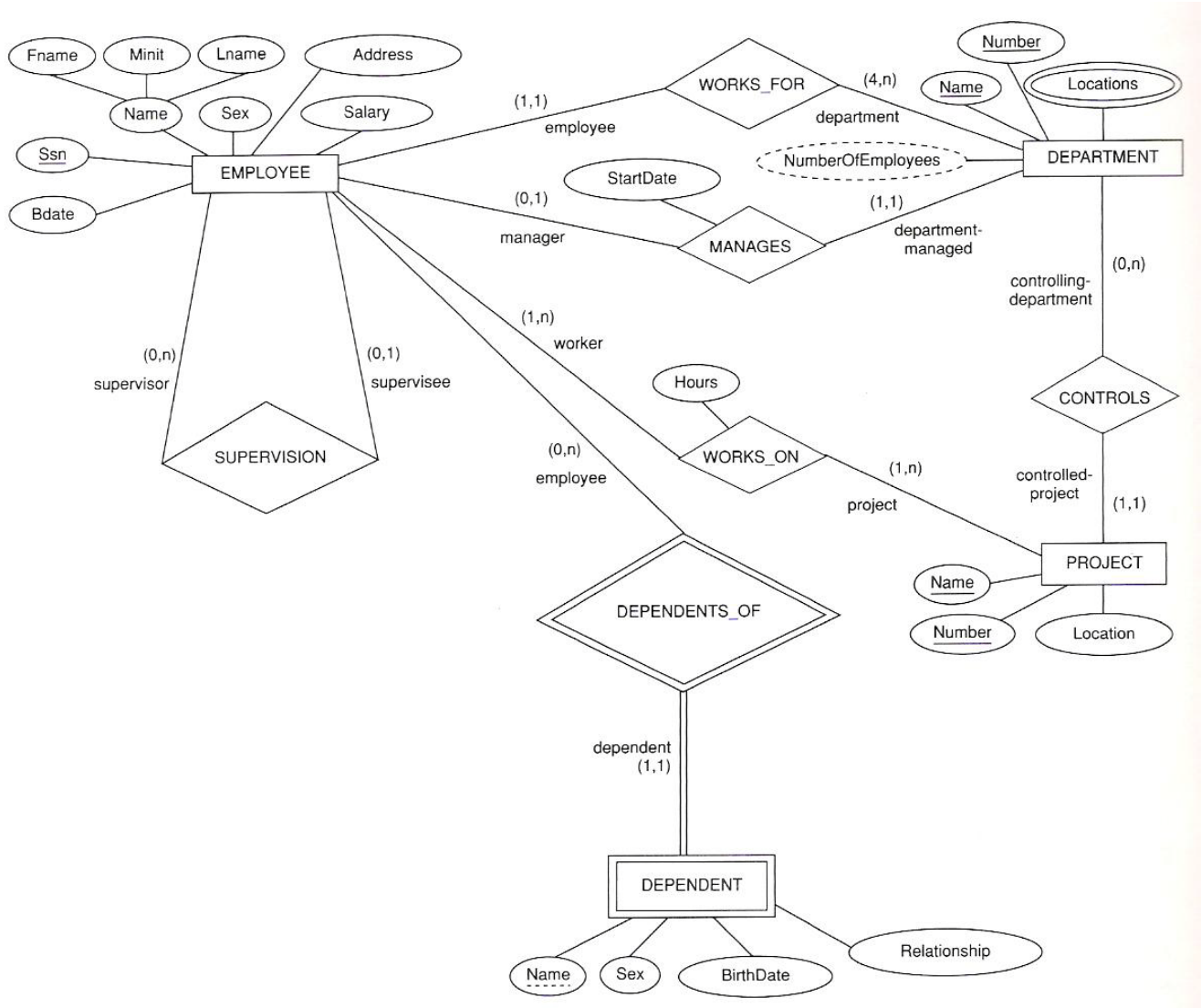

Figure 2. ER diagram for a company database

The company aspects that are represented in the database are described below:

- The company is organized into departments. Each department has a name, a number, and an employee who manages the department (with a corresponding start date when that employee started managing the department). A department may have several locations.
- A department controls a number of projects; each project has a name, a number, and a single location.
- Each employee's name, social security number, address, salary, sex, and birth date are stored in the database. An employee is assigned to one department but may work on several projects, which are not necessarily controlled by the same department. The database keeps track of the number of hours per week that an employee works on each project, and of the direct supervisor of each employee.
- It is necessary to keep track of the dependents of each employee for insurance purposes. The following information is required: each dependent's name, sex, birth date, and relationship to the employee.

Assignment Steps

a) Domain Model Definition

Design the domain model containing the concepts needed to express any ER diagram by following the notation introduced above. Use the structural constraint notation to express constraints on relationship types.

What to hand in:

- UML diagram(s) for the domain model.

b) UML Profile Definition

Map the ER domain model to a UML profile.

What to hand in:

- UML profile diagram containing the profile definition.
- A brief textual description of your mapping strategy.

c) Apply the UML profile to represent ER models

In this step, you are asked to represent in UML extended with the ER profile defined in the previous step two ER examples: a) the first is the ER model from figure 2, and b) the second is an ER model of your choice.

What to hand in:

- The diagrams for the respective examples expressed in UML extended with the ER profile you have defined. You may add comments in notes to the UML diagram to indicate the values of the stereotype attributes if the tool you are using does not display them. For the second example, include also the ER diagram and a brief explanation of what it represents.

==========