Extended Entity-Relationship (EER) Modeling

Peter Y. Wu
Department of Computer and Information Systems
Robert Morris University

Extended Entity-Relationship Model

- **EER** – Extended or Enhanced ER Model
- Developed in the mid 1980s… bearing also the influence of object-oriented modeling.
- The main concept is: **Generalization** and **Specialization** – one and the same concept.
- Other concepts such as Categorization and Aggregation will not be covered; these are not considered main concepts.

EER: Generalization/Specialization

Consider modeling the employees of a company, such as a personnel database system. We need to have name, employee id, address, department. (as attributes for each employee…)

- Some employees have a monthly salary, but some others have an hourly wage rate and are paid by the hours they have worked. How should we then model that?

EER: Generalization/Specialization

- Let us try two kinds of employees: salaried-employee, and hourly-employee …
- Not a good solution: because we need to support the idea of employee – for applications such as sending a letter to each employee; should not need to deal with two kinds.

SOLUTION: Specialization

- Define a subset of an entity set, called a subclass.
- Establish additional attributes for the subclass.
- Subclass entities inherit attributes from the superclass.

EER: Specialization

- Employee is an entity type, with these attributes: name, employee id, address, department…
- **Specialization:**
  - Hourly Employee is a subclass of Employee.
  - Each entity of Hourly Employee is also an entity of Employee, but has the additional attribute: hourly rate.
  - Employee is the superclass of Hourly Employee.
  - New ER Diagram symbol:
EER: Subclass/Superclass

Formal definition:
For every entity \( e \in E2 \Rightarrow e \in E1 \).

- That is, every entity of the subclass \( E2 \) is also an entity of the superclass \( E1 \).
- Thus, \( E2 \) inherits every attribute of \( E1 \), and every relationship involved with \( E1 \).
- Key attributes for \( E1 \) also serve for \( E2 \).

Since \( E2 \) inherits attributes from \( E1 \), we cannot define an attribute for \( E2 \) with the same name as an attribute of \( E1 \).

A similar rule applies to relationships.

The key defined for \( E1 \) applies also to \( E2 \); we cannot define another key for the subclass. (But possibly an alternate key.)

E1

---

E2

---

\[ \forall e \in E2 \Rightarrow e \in E1 \]

That is, every entity of the subclass \( E2 \) is also an entity of the superclass \( E1 \).

Thus, \( E2 \) inherits every attribute of \( E1 \), and every relationship involved with \( E1 \).

Key attributes for \( E1 \) also serve for \( E2 \).

EER: Specialization example

Every hourly employee (\( e \in E2 \)) is also an employee (\( e \in E1 \)).

- Employee
  - name
  - employee id
  - address
  - department
  - hourly rate

Hourly Employee

- Dn

Specialization:
- If an employee is related to some projects (to work on), the same applies to an hourly employee.
- Hourly employees may also work on some projects.

Employee

---

Project

Employee

---

Project

EER: Specialization Constraints

A salaried employee is NOT an hourly employee, and an hourly employee is NOT a salaried employee; the subclasses are disjoint.

Employee

---

Salaried Employee

---

Hourly Employee

- \( E1 \)
- \( E11 \)
- \( E12 \)

\( E11 \) and \( E12 \) are disjoint ...

- For every entity \( e, e \in E11 \Rightarrow e \in E12 \) and \( e \in E12 \Rightarrow e \in E11 \).

That is, \( E11 \cap E12 = \emptyset \).

EER: Specialization Constraints

Every employee is either salaried or hourly; the superclass entity employee has total participation in the specialization.

Employee

---

Salaried Employee

---

Hourly Employee

- \( E1 \)
- \( E11 \)
- \( E12 \)

\( E1 \) has total participation in the specialization into \( E11 \) and \( E12 \).

- For every entity \( e, e \in E1 \Rightarrow e \in E11 \) or \( e \in E12 \).

That is, \( E1 \cup E12 = E1 \).
**EER: Specialization Constraints**

- Some employees are technicians but not every one; the superclass entity employee has **partial participation** in the specialization.

**Diagram:**

```
Employee
  ↓
Technician
  ↓
E1
```

- **E1** has **partial participation** in the specialization into **E2** ...

- There may exist some entity **e, e ∈ E1** such that **e ∈ E2**.

**Diagram:**

```
Employee
  ↓
Salaried Employee
  ↓
Technician
  ↓
E1
  ↓
E11
```

- **E11** and **E12** are **overlap** ...

- There may exist entity **e, e ∈ E1** such that **e ∈ E11** and **e ∈ E12**.

- That is, **E11 ∩ E12 ≠ Φ**.

**EER: Specialization Constraints**

- A salaried employee may also be a technician; the two entity sets – **subclasses** of entity set Employee **overlap**.

**Diagram:**

```
Employee
  ↓
Salaried Employee
  ↓
Technician
  ↓
E1
  ↓
E11
```

- **E11** and **E12** are **overlap** ...

- There may exist entity **e, e ∈ E1** such that **e ∈ E11** and **e ∈ E12**.

- That is, **E11 ∩ E12 ≠ Φ**.

**EER: Specialization Constraints**

- Specialization constraints NOT specified...

**Diagram:**

```
Employee
  ↓
Salaried Employee
  ↓
Technician
```

- Quite **possibly** the two subclasses **overlap**, but the designer did not specify. The two subclasses were derived from two different paths of specialization.

**EER: Specialization Constraints**

- Specialization of an entity set leads to subsets of the entity set; these are called **subclasses** of the **superclass**.

- The participation of the **superclass** in the **specialization** may be **total** or **partial**.
  (Do not confuse with participation constraint in a relationship!)

- The **subclasses** in the specialization may be **disjoint** or **overlap**. (Note: overlap may lead to complications.)

**EER: Specialization Constraints**

- A subclass as an entity set can be specialized further into other subclasses.

- But overlapping subclasses may specialize into the same subclass entity set, having multiple superclass entity sets.

- Consider the intersection of two overlap subclasses...

- It is a subclass which **inherits** from **more than one** superclass, attributes as well as relationships.

- It is called **multiple inheritance**.
EER: **Specialization**

Example of multiple inheritance...

- **university member**
  - **social sec no.**
- **employee**
  - **wage rate**
- **student**
  - **class**
- **student assistant**

EER: **Generalization**

- For specialization, we start with the superclass.
- Specialization creates subsets – subclasses of the superclass.
- **Generalization** is the reverse of specialization, we start with the subclasses.
- Generalization formulates a collective way of characterizing the concept of all subclasses as an entity set, that is, the superclass.

EER: **Generalization** Example

Given trucks, and cars, as two entity sets...

- **Vehicle**
  - **license no.**
- **Truck**
  - **license no.**
- **Car**
  - **max passengers**

• Generalization provides the same modeling functionalities as specialization. No difference!

Understanding EER Model ...

• For every entity \(e_2\) of \(E_2\), \(e_2\) is also an entity of \(E_1\). Therefore, \(e_2\) has values for all these attributes \(A_1, K, A_2\).
• Attribute \(K\) is key for \(E_1\); it can also serve as a key attribute for \(E_2\).
• However, an entity \(e_1\) of \(E_1\) may or may not be an entity of \(E_2\).