CS 320
Introduction to Software Engineering
Spring 2017

February 15, 2017
Recap: the analysis phase

- It is the process of formally specifying exactly **what** is to be built based on the client’s ideas and needs.
- Obtain a **clear understanding** of the **client’s needs** (clear means unambiguous in this context).
- **Software engineer** is an **active participant**. Client may not understand the potential/feasibility of a system.
- Analysis **may uncover** unknown potential/**pitfalls**.
Recap: the analysis phase

The high-level questions to answer
1. **What is the purpose** of the system?
2. **Who uses the system** (can be human and non-human)?
3. **What functionality/benefits** must it provide (what’s the desired behavior)?
4. **Is it feasible** to build the system with the given expectations?
Today

Post-mortem of discussion session
● What analysis techniques did you use?
● How useful/informative was the client interview?
● What are the lessons learned?
● How will you proceed?

Modelling techniques
● Entity-Relationship (ER) diagrams
Open discussion

Post-mortem of discussion session
● What analysis techniques did you use?
● How useful/informative was the client interview?
● What are the lessons learned?
● How will you proceed?

The high-level questions
● What’s the purpose of the system?
● Who uses the system?
● What functionality must it provide?
● Is it feasible to build the system?

Common analysis techniques
● Meetings, questionnaires, interviews.
● Prototyping.
● User stories, use cases.
● Modeling (Diagramming).
ER diagrams: overview

- An Entity Relationship (ER) diagram is a **graphical representation** of a **data model**.
- It shows the **relationship** between **entities** (e.g., people, objects, events, or concepts) within a system.
- It can be mapped to a relational (database) schema.
ER diagrams: graphical syntax

- An entity $E$
ER diagrams: graphical syntax

- An entity $E$

- An attribute $A$ of entity $E$
ER diagrams: graphical syntax

- An entity $E$
- An attribute $A$ of entity $E$
- A relationship $R$ between two entities $E_1$ and $E_2$
ER diagrams: graphical syntax

- An entity $E$

- An attribute $A$ of entity $E$

- A relationship $R$ between two entities $E_1$ and $E_2$

- An attribute $B$ of relationship $R$
ER diagrams: rules

- An interconnecting line is only allowed between:
  - a box and a diamond,
  - a box and an oval,
  - a diamond and a oval.

- An oval must have exactly one connecting line.

- Names of boxes must be unique in the diagram.

- Names of ovals must be unique per box/diamond.
ER diagrams: example

Let’s model the following entities and their relationships in the context of a simple course registration system at UMass:

- Students
- Instructors
- Courses
ER diagrams: example

- **Student**
  - SpireID
  - Name
  - Takes
- **Course**
- **Instructor**
  - SSN
  - Name
  - Teaches
A key is an (underlined) attribute, or a set of attributes, which uniquely identifies an entity.
ER diagrams: keys and cardinalities

- A key is an (underlined) attribute, or a set of attributes, which uniquely identifies an entity.
- A key can be **artificial** or **natural**.

![ER diagram](image-url)
ER diagrams: keys and cardinalities

- A key is an (underlined) attribute, or a set of attributes, which uniquely identifies an entity.
- A key can be artificial or natural.
- The cardinalities define the kind of relationship (one-to-one, one-to-many, or many-to-many).
- There are different notations for cardinalities. For example:
  - $1 = (1,1)$
  - $c = (0,1)$
  - $m = (1,*)$
  - $mc = (0,*)$
A weak entity can’t exist on its own (if a building is torn down, its rooms disappear).
ER diagrams: weak entities

- A weak entity can’t exist on its own (if a building is torn down, its rooms disappear).
- A weak entity is only uniquely identifiable in reference to another entity.
ER diagrams: generalization

- An `is_a` relationship represents a generalization relationship between two entities.
ER diagrams: generalization

- An is_a relationship represents a generalization relationship between two entities.
- Attributes (including keys) are “inherited”.

![ER Diagram]

- Person
  - SSN
  - Name
- Student
  - GPA
- Instructor
  - SRTI
ER diagrams: generalization

- An is_a relationship represents a generalization relationship between two entities.
- Attributes (including keys) are “inherited”.
- Additional attributes can be defined.
ER diagrams: self references and roles

- A **self reference** is usually explicitly annotated with **roles** to clarify the meaning of the self-referencing relationship.

```
Employee manages
| is managed by |
```

Think about (but never draw) the following:

```
Employee manages
| is managed by |
Employee
```
ER diagrams: example

Let’s improve our initial model of a simple course registration system at UMass:
● Students
● Instructors
● Courses
● Sections
● Prerequisites
● Assignments
● Points/grades
ER diagrams: example

**Student**
- SpireID
- Name
- Grade

**Instructor**
- SSN
- Name

**Course**
- teaches
- prerequisites for
- requires

**Assignment**
- submits
- MaxPoints
- A_Id

**Section**
- Sec_Id

**Relationships**
- Student takes Course
- Instructor teaches Course
- Course has Assignment
- Assignment submits MaxPoints
- Course has prerequisites
- Course has prerequisites for
- Course requires
- Course requires
- Student A_Id
- Instructor SSN