Notes: Entity-Relationship Diagrams

After a systems requirement have been captured the design work can begin.

Designs often need to be shared and discussed with many stakeholders such as business execs, users and developers. The ERD is a non-technical tool for communicating design free of ambiguities. An ERD is a type of diagram which shows how entities within a system relate to one another.

There are different ways of representing/drawing ERD's. The UML specifies a specification that is widely used.

Components of Entity-Relationship Diagrams

There are 3 primary component types in an ERD

1. Entities
2. Attributes
3. Relationships

Sub Components of ERD's

Entity Types

An entity type represents a group of objects with the same attributes.

A Strong Entity is an entity which is existence is not dependent on another entity type.

A weak entity is an entity that cannot exist by itself, it needs a relationship to another entity to make it identifiable.

An entity instance represents a specific instance on an entity type.

Relationships

A relationship type defines the relationship between entity types. A semantic net can be drawn to show the relationship between specific entity instances. Relationships can occur between multiple entities. A relationship between 2 entity types is called a binary relationship. A relationship between 3 types a tertiary relationship.

Relationships between more than 2 entity types are considered complex relationships. In the UML specification, a diamond is used to show a complex relationship.

A relationship can have attributes that capture specific details. In UML these attributes are also drawn in a box similar to an entity. A dashed line is used to connect the attributes to the relationship, this differentiates it from an entity.

A role can be assigned to describe the role an entity has in a relationship. For example, a relationship between a 'Staff' entity type and a 'Branch' entity type may assign a role of ‘manage’ to a relationship.

Attributes

The properties of entity types are called attributes. Attributes can have domains. A domain is the bounds for which the attribute must exist within, for example, $n > 5$.

A **multivariable** attribute is an attribute that can have multiple values, i.e. a person may have multiple phone numbers.

A **derived** attribute is an attribute that is derived based on other attributes. For example, a person's age is derived of their date of birth and the current date.

Keys

A **candidate** key is a set of attributes that uniquely identify an entity.

The **primary** key is the candidate key that is used to identify an entity.

A **composite** key is a candidate key with multiple attributes.

Generalization.

Similar to Class Diagrams or in OOP subclasses can inherit attributes from superclasses.

Total participation

For total participation of an entity set to be achieved every entity must have a relationship to at least one other entity i.e. no entities are 'floating' unrelated.

The various levels of models

An ERD can describe a system at various levels of detail.

The three main layers detailed are:

The conceptual model- This level shows the least detail but clearly describes the scope of the system and its key concepts

The Logical data model - This level contains more details and attributes but could not yet be used for implementation due to lack of tech-specific detail.

The physical data model - This level contains all the data required to implement the system's database using real technology