Entity Relationship Modelling with Visual Paradigm

1. Introduction

This guide is intended to introduce you to the process of creating Entity Relationship Diagrams using Visual Paradigm and UML. If you have not used Visual Paradigm before it is recommended that you first complete the “Brief Introduction to Visual Paradigm for UML” worksheet available on Moodle.

Entity Relationship Modelling is a modelling technique used to describe the data or information elements of a business system. The main output of the modelling technique is an Entity Relationship Diagram or ERD. Entity Relationship Modelling is widely used to create a model of an information system that can be used as a design specification for a relational database. Therefore, ERDs are commonly associated with database design.

2. An Important Note on Notation

Although the core concepts of ERDs have remained relatively constant, over the years there have been several different ERD notations (i.e. sets of symbols, diagram layouts and rules). You are likely to encounter more than one notation when working in the industry depending on the age of the modelling material in the organisation. There has been a general move towards UML in the software engineering industry and, although designed for object-oriented software engineering, UML Class Diagrams are conceptually equivalent to ERDs in that they show abstract ‘things’ (i.e. classes/entities) which are identified in the ERM process.

We will therefore use Visual Paradigm’s UML Class Diagram tool to create our ERD but will only use a selection of the relevant Class Diagram elements. For reference, a mapping of terms between ERD and UML Class Diagram notation is included below in table 1.

Entity Relationship Diagrams	UML Class Diagrams
Entity	Class
Relation	Association
Cardinality	Multiplicity
Optionality	

Table 1: ERD <> Class Diagram notation mapping

Note that Visual Paradigm does have a separate Entity Relationship Diagram tool which uses a notation known as ‘crow’s foot’ or Barker’s notation. You may wish to try this tool in your own time to familiarise yourself with the notation but bear in mind that this notation may be considered outdated in the industry so in the Information Systems unit we use the class diagram-like notation.

3. Creating an Entity Relationship Diagram

Create a new Entity Relationship Diagram (ERD) using the UML Class Diagram type by either selecting “Information System (UML) > Class Diagram” from the Start Page, or by selecting “Diagram > New > Class Diagram”. You will be asked to name your diagram.

A blank canvas will appear with a Class Diagram specific toolbox just to the left (see Figure 1). The main diagram elements that we will use are Classes (to represent Entities) and Associations (to represent Relations: one to one, one to many, many to many). Most of the other diagram elements are relevant only to Class Diagrams, not ERDs.

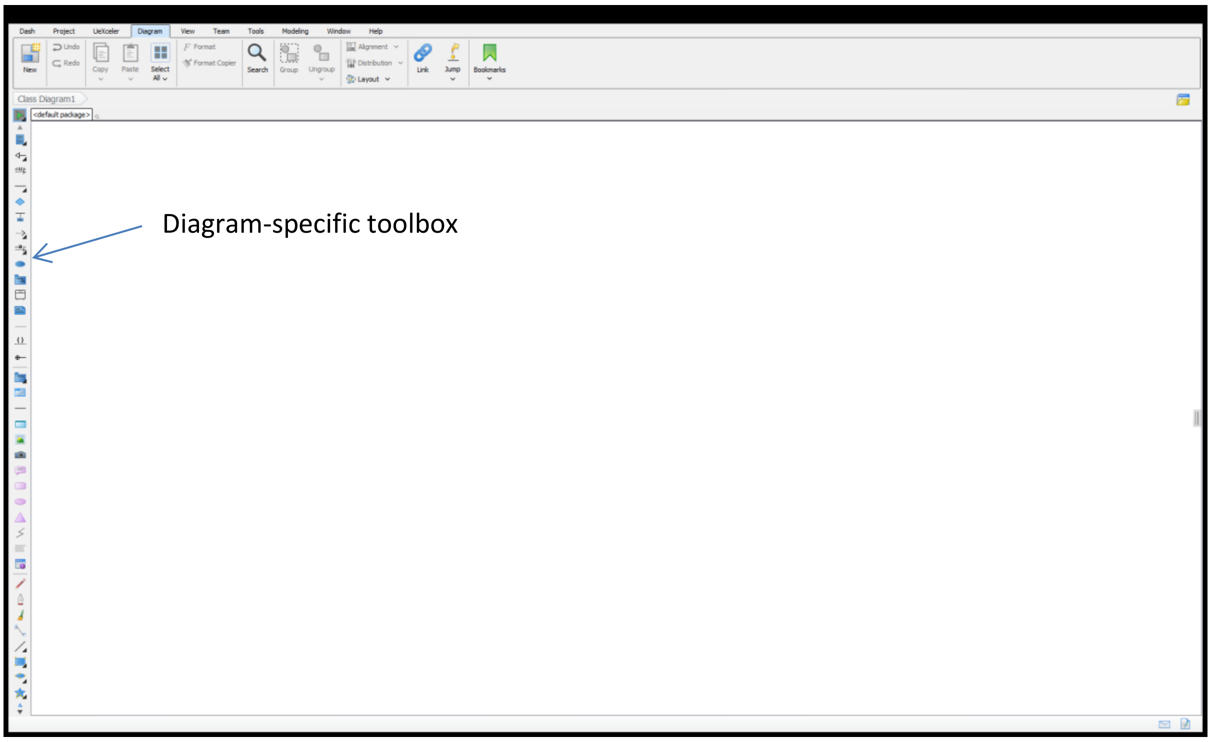


Figure 1: Blank Entity Relationship Diagram canvas

As with all Visual Paradigm diagrams you can rename the diagram by right clicking the 'Diagram Name' in the top left hand corner of the canvas. In the case of Class Diagrams, Visual Paradigm will also create a Package. We will not be making use of Packages in the creation of our ERD but you can change the name of both the Package (double-click to rename) and the Diagram to "Student Records System".

4. Adding Entities to the Diagram

First we'll add an entity named 'Student' to our diagram.

Click 'Class' from the toolbar on the left (see Fig. 2) and then click on the canvas where you want to place the new entity.

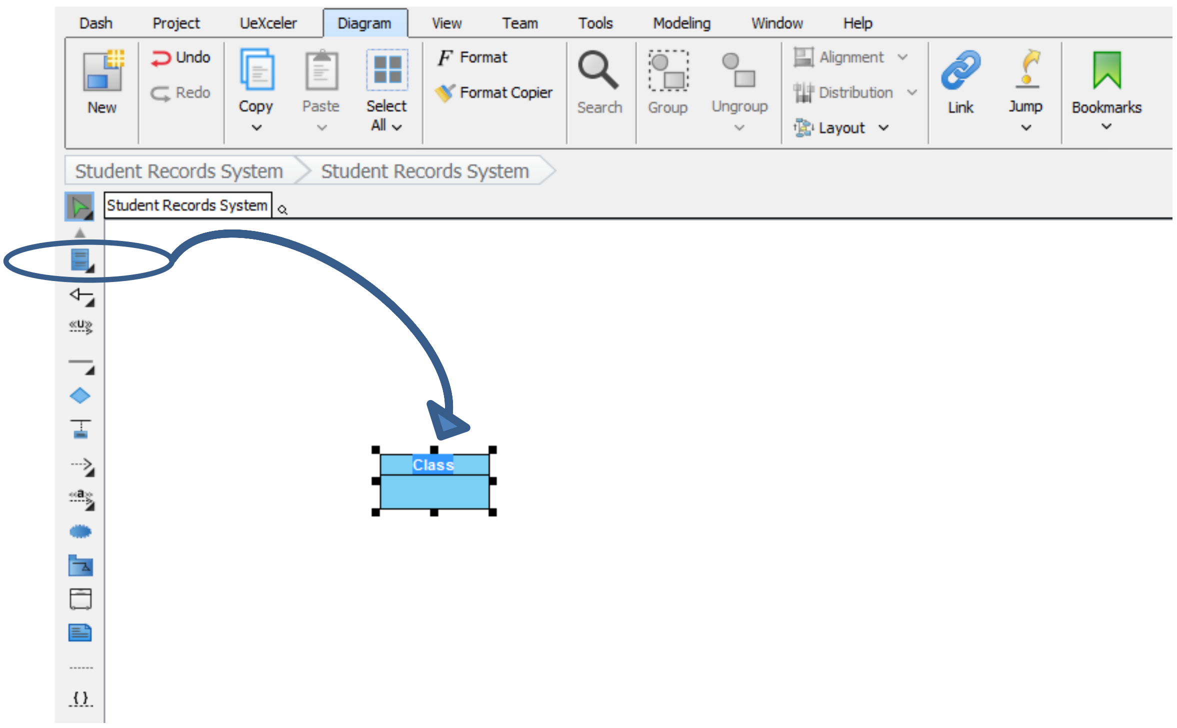
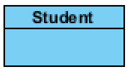


Figure 2: Creating an Entity

Once placed on the canvas you can type the name of the new entity. In this case name it 'Student' and the press enter.

Once created the entity should be shown like this:



Repeat this process to add another entity called 'Tutor'.

5. Adding Relationships between Entities

You should now have a diagram with two entities as shown in Figure 3. We need to represent the relationship between students and their personal tutors. We will assume that this is a 'one to many' relationship i.e. a student has a single personal tutor and a tutor can supervise many students. We will also assume that a student must have a personal tutor but that some tutors may not supervise any students. Therefore, another way to describe this relationship is that a student has a minimum of 1 tutor and a maximum of 1 tutor, and that a tutor has a minimum of 0 students and a maximum of many students.

Student
Tutor

Figure 3: Student & Tutor Entities

We create relationships by using the Visual Paradigm 'Association' diagram element. As with other diagrams this can be achieved by clicking on the — Association button on the toolbar and then dragging a line between the two entities.

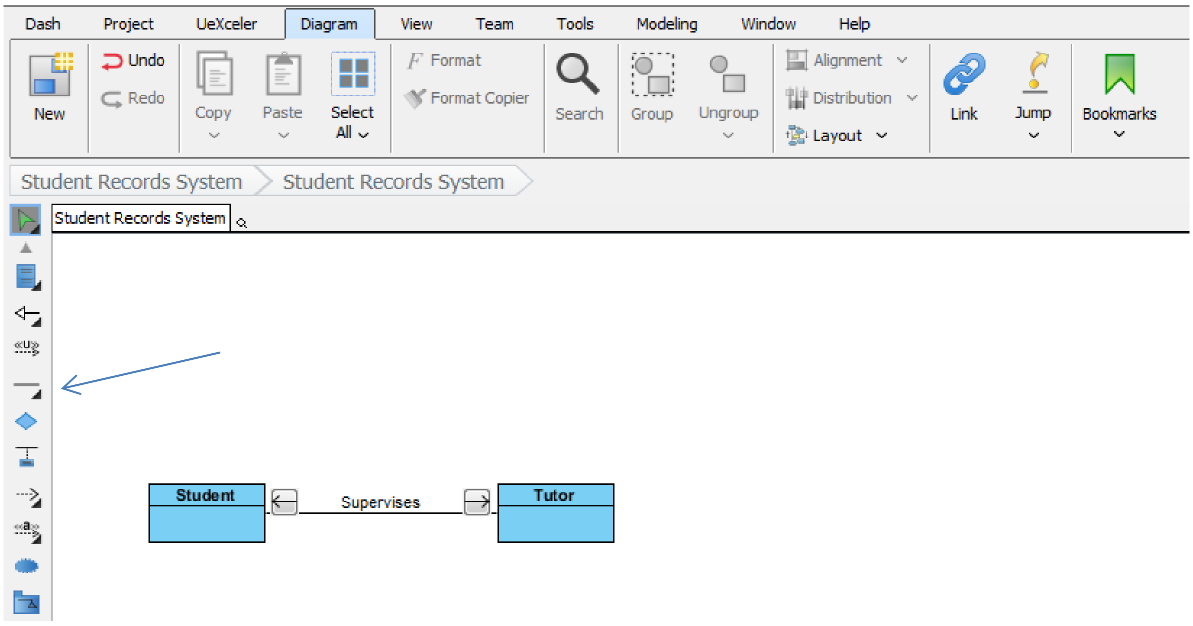
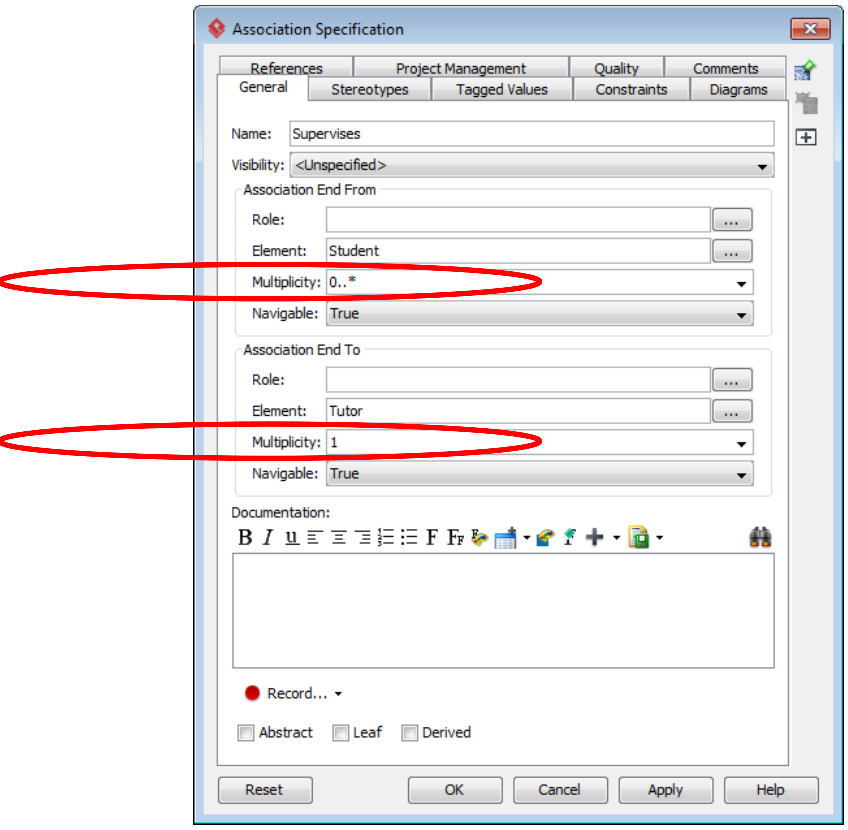


Figure 4: 'Supervises' Relationship

Create an association between Student and Tutor. Once you have created it, you can name the association. Type the label 'Supervises' to indicate that the tutor supervises students (see Figure 4).

Next we need to add details about the numbers involved in the relationship. This is referred to as cardinality and optionality in Entity Relationship Modelling and multiplicity in UML.



Name: Supervises
Visibility: <Unspecified>
Association End From
Role:
Element: Student
Multiplicity: 0..*
Navigable: True
Association End To
Role:
Element: Tutor
Multiplicity: 1
Navigable: True

Figure 5: Association Specification

There are two main ways of setting this up. The first way is to right-click on the association and select 'Open Specification'. This will bring up a properties window. Under the 'General' tab you will see two sections 'Association End From' and 'Association End To' which represent the two ends of the association. Notice how one is labelled 'Element: Student' and the other 'Element: Tutor' so you can easily distinguish between them.

Change the Multiplicity to 0..* at the student end of the association, and to 1 at the Tutor end as shown in Figure 5. In other words a student must have 1 and only 1 tutor, and a tutor can supervise a minimum of 0 and maximum of many students.

The alternative method to set multiplicity is to right click on the association on the diagram and select Multiplicity in the menu. Note that if you right click in the centre of the association line you will be shown two menu entries for each end of the association (in this case Role A (Student) and Role B (Tutor)) as shown in Figure 6. However if you right click on the end of an association line you will only be shown the multiplicity settings for that end of the relationship.

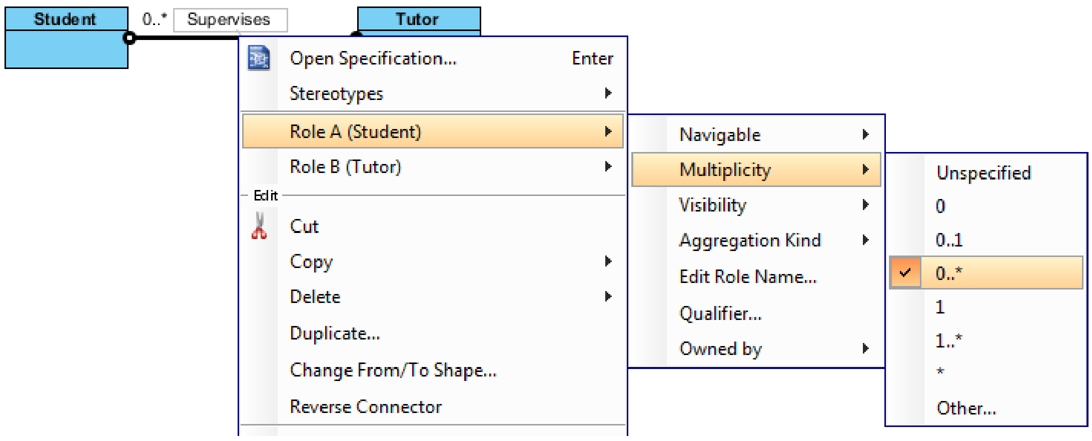


Figure 6: Alternative method to set multiplicity

Once you have set multiplicity using one of the above methods you should have a diagram like the one shown in Figure 7.

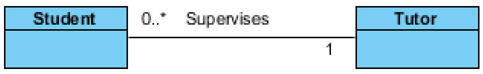


Figure 7: Entities with Relationship

Although this is a very simple system that we have modelled, the technique is the same for all other entities and relationships that you may wish to add.

During the initial stage of entity relationship modelling it is common to create a conceptual model in this manner or, more likely, several iterations of the design in order to identify all possible entities and relationships and to get a general understanding of the system design. However, once the entities and relationships have been defined it is then useful to add more detail.

6. Adding Attributes to the Entities

We will now add some attributes to the Student entity. This will allow us to design what data will be kept about each student and to specify primary and foreign keys. We can also optionally specify which datatypes each of the attributes will use.

Right-click on the Student entity, go to ‘Add’ and then select ‘Attribute’. This will add a new attribute to the Student entity. Type ‘studentNumber’ (note attributes should start with a lower case letter, contain no spaces and that each subsequent word should be capitalised) then press Enter. Visual Paradigm will create the attribute and then add another line for you. Continue to add these attributes:

studentName
studentAddress
studentPhoneNumber
studentTutor

When you have finished adding the attributes press Escape to stop adding. If you need to edit any of the attributes just double click on them.

We now need to set the studentNumber attribute as a primary key and studentTutor as foreign key. To do this we will use UML Stereotypes. The primary key Stereotype (PK) already exists in Visual Paradigm but we will need to add a foreign key (FK) stereotype.

Right click on the studentNumber attribute, go to ‘Stereotypes’ and select ‘PK’ from the menu. Visual Paradigm will add <<PK>> to the beginning of the attribute name to represent the primary key.

Now we need to add the foreign key stereotype. Right-click on the studentTutor attribute, go to ‘Stereotypes’ and the select ‘Edit Stereotypes…’. In the dialog box that appears click the ‘Edit Stereotypes…’ buton. Next click ‘Add’ to create a new stereotype. Name the stereotype FK and click OK. Click OK twice more to return to the main canvas.

As you did with the primary key, you should now be able to right click on the studentTutor attribute, go to ‘Stereotypes’ and select ‘FK’ from the menu. Visual Paradigm will add <<FK>> to the beginning of the attribute name to represent the foreign key.

You should now have a diagram that looks like the one shown in Figure 8.

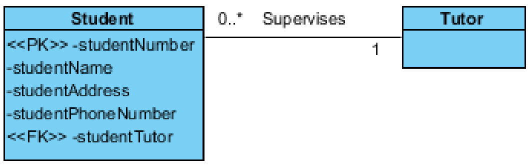
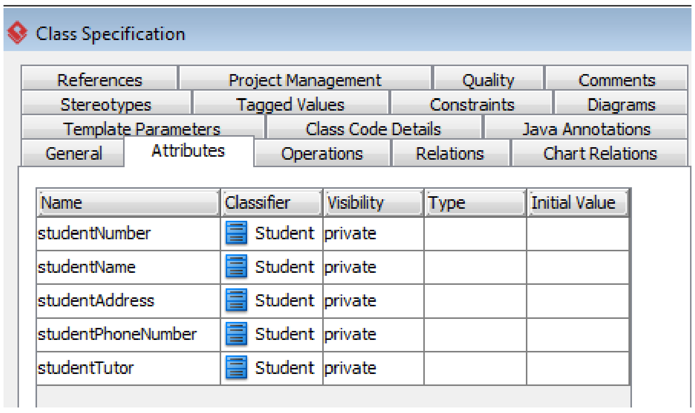


Figure 8: Diagram with Student Attributes

7. Alternative Method for Adding and Editing Attributes

An alternative way of adding attributes or for editing them at a later date is to use the attribute specification dialog. Right-click on the Student entity (make sure you click on the top section of the entity where the Student label appears) and select 'Open Specification' from the menu.



Name	Classifier	Visibility	Type	Initial Value
studentNumber	Student	private		
studentName	Student	private		
studentAddress	Student	private		
studentPhoneNumber	Student	private		
studentTutor	Student	private		

Figure 9: Attribute Specification

Then go to the 'Attributes' tab and you will see the attributes that you have just created (as in Figure 9). From this screen you can add new attributes or open the specification of individual attributes. You can also specify the type of the attribute such as integer, Boolean, character, string etc. At this stage it is not necessary for you to define the types but you should be aware that they can be specified.

To be properly compliant with UML an entity's attributes should also be set to public. This can be set in the 'visibility' column shown in Figure 9. Public attributes will be displayed with a '+' in front of their name rather than the '-' shown in front of private attributes. In practice, this is important for software development but inconsequential for database design.

8. Finishing Off

You can now finish off the diagram yourself. Add appropriate attributes to the Tutor entity and define an appropriate primary key. Remember that the primary key of the Tutor entity will relate to the 'studentTutor' foreign key of the Student entity. Also feel free to add additional entities and relationships for additional practice.