CECS 323 LAB DIA TIPS AND TRICKS

OBJECTIVE: Give you a quick introduction to the DIA drawing tool so that you can do your modeling.

INTRODUCTION: It will quickly become evident to you that DIA is not enforcing any of the business rules of the UML modeling languages when you build these diagrams. That has to be up to you.

Both the graphical languages that we will introduce you to are used for application design and communicating that design. Every model concentrates on some aspects of what it depicts and suppresses/abstracts the rest. In our case, we concentrate on the parts of the classes that we can represent in the relational database management system.

In the relation scheme diagram (as you will learn later) the UML diagram content is augmented/built upon to capture various keys in the tables used to represent the classes, and how those keys are used to implement the associations between the classes. Rather than try to get all this design done at once, we approach it in a stepwise fashion so that we can identify and resolve issues at each level before committing ourselves to a suboptimal design.

The relation scheme diagram is a proprietary graphical design language built here at Long Beach State. It serves the need well, as a bridge between the UML model and the physical model (the actual table design, which you will see later). That is the good news. The bad news is that since relation scheme diagrams are rather specialized, there is very little tool support specifically for that graphical language, so we have to use generic graphical objects to represent what we need in the relation scheme diagram.

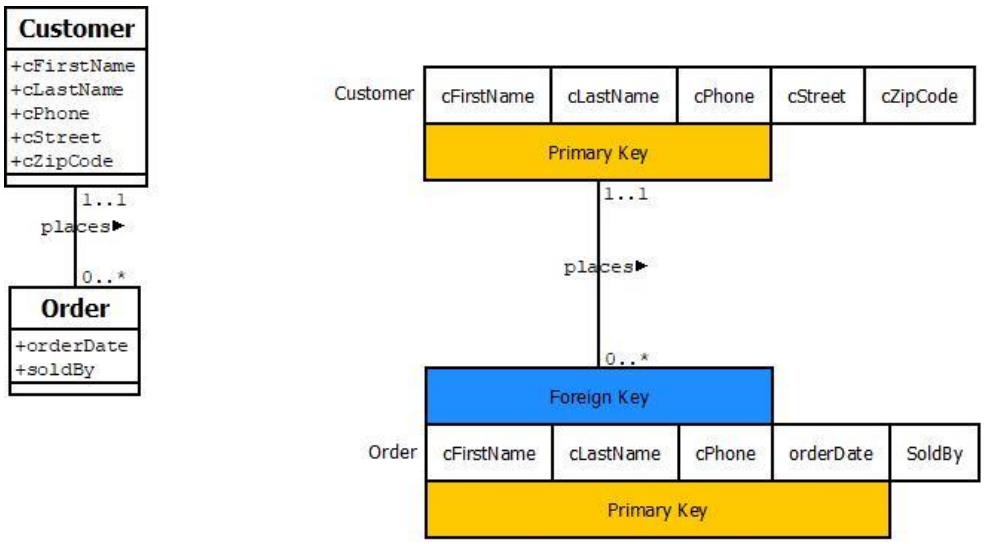
There are several tools that we could use for diagraming. In this lab you will get an introduction to DIA. You will also get introduced to drawio.net. They are both free, and they get the job done.

PROCEDURE:

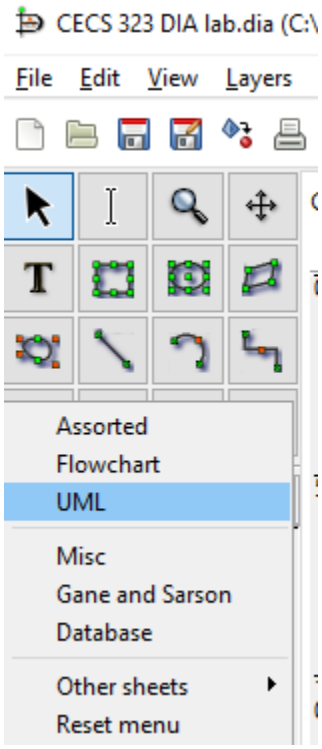
At this point in the semester, don't get too worried about the concepts behind the diagrams, we will be introducing you to those along the way. But I had to do some samples that were complex enough that you could see all the ingredients to the models that you're likely to be doing in this class. I will, however, be going over a few modeling guidelines and standards because I've found that I need to repeat those before they sink in.

We are going to walk you through the steps necessary to build this UML class diagram and relation scheme diagram. The UML class diagram is the one to the left.

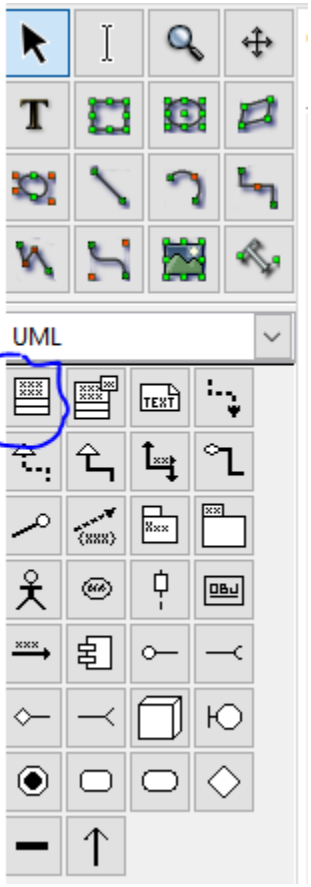
CECS 323 LAB DIA TIPS AND TRICKS



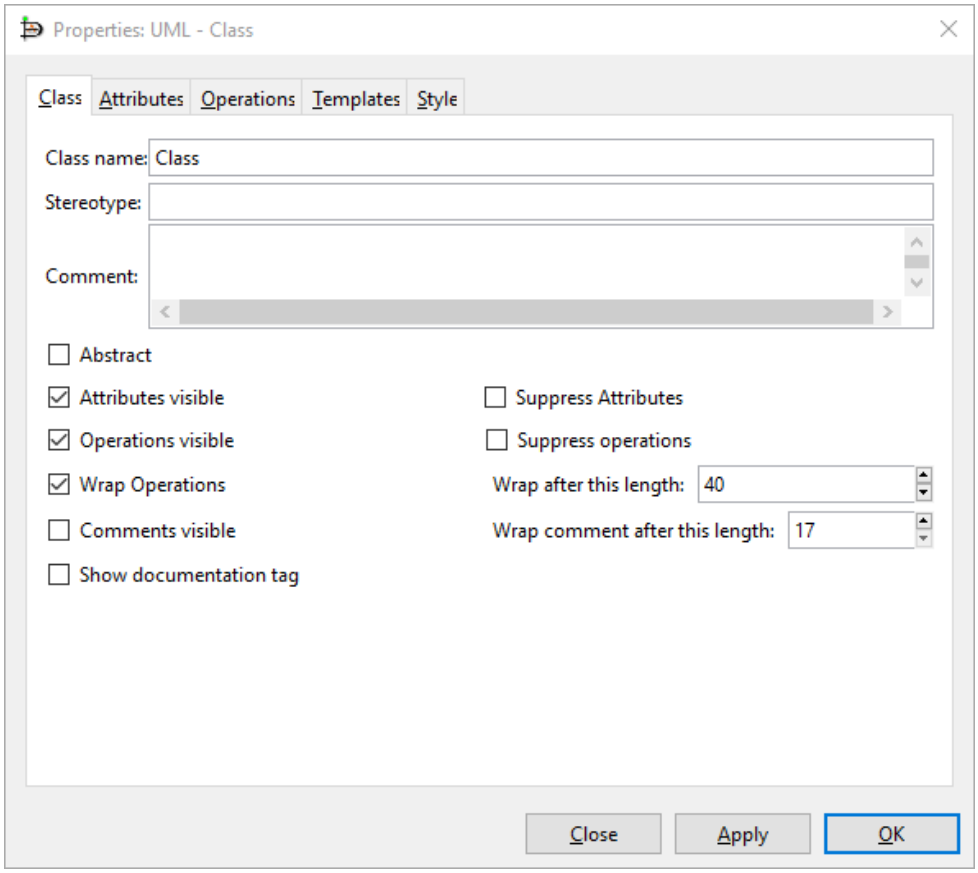
1. For starters, you can download DIA and install it from: http://sourceforge.net/projects/dia-installer/.
    a. For my MAC friends, try http://dia-installer.de/download/macosx.html.en. I don't have an easy way to test this distribution of DIA, but it looks promising on the web anyway.
2. We're going to start with the UML diagrams, since typically you build those first.
    a. Select the UML stencil from the pulldown:



    b. The class object is the upper left-hand corner of this stencil.



c. Drag that object from the stencil to where you want to put the upper left-hand corner of your class on the canvas.
d. Override the class name (it defaults to “Class”).
    i. Double click the new class that you have just positioned in your model.
    ii. A dialog box appears that looks like:

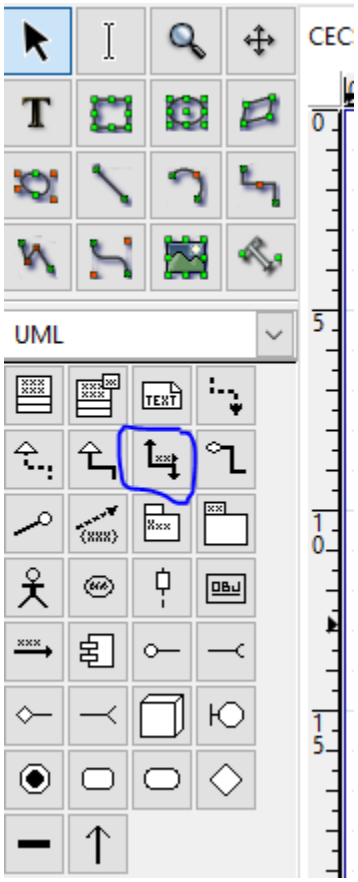


iii. Select the "Class" in the "Class name:" field and give your class an appropriate name. In this case, we will start out with "Customer" for our first class.

e. Then use the Attributes tab to enter the attribute(s), hitting "new" each time you get to a new attribute.

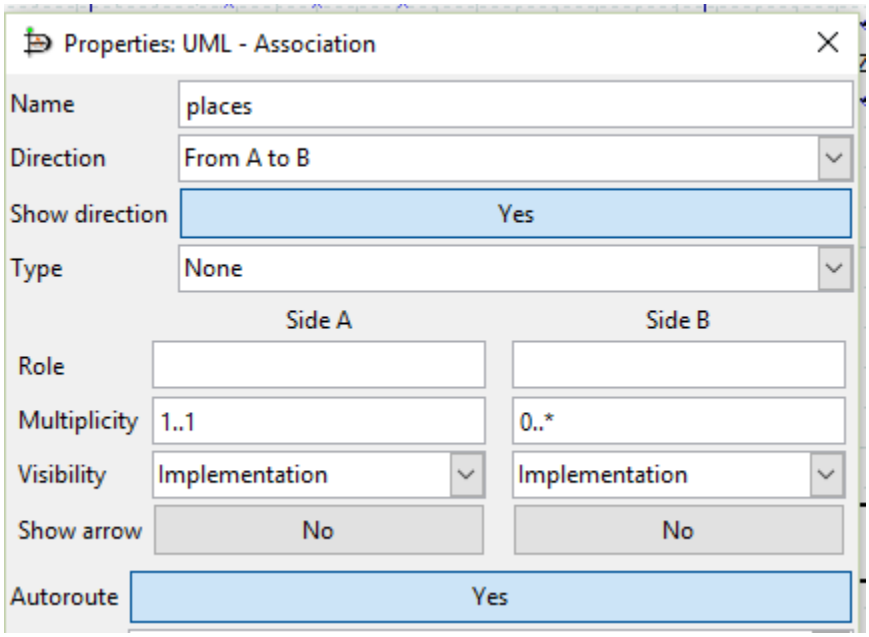
f. Use the same approach to create a second class, called "Order".

g. The association **between** two classes is:



h. Click the association icon, then click one of the attach points on the parent class (Customer in this case). That will yield an association that points to nowhere. Note that the attach points on the class look like tiny "x" marks around the edge of the class. One frustrating aspect of DIA is that the classes only have so many attach points. Please do not start or terminate more than one association from the same attach point on a class. That is invalid graphical syntax.
    i. Drag that **child** (the one that is not attached to anything) end of the association over to the child class and attach it at an attachment point.
    ii. If the association is not attached, when you move the class, the association terminus will not follow. Rather irritating at best.

i. Then fill in the metadata for the association:



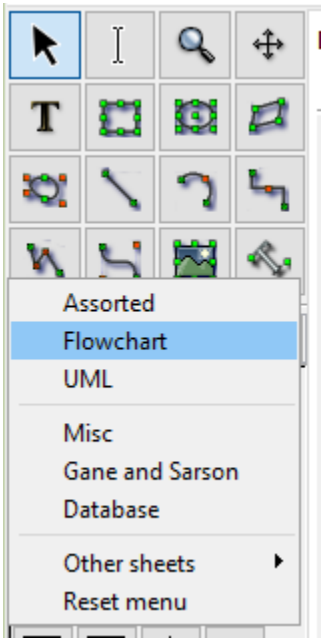
i. The name is the "verb phrase" for the association.
   1. You **can** have a forward and a reverse verb phrase, and we will use that on occasion in this class, but generally just the forward phrase is enough, which means that the Side A and Side B roles don't have to be specified.
   2. Note that the multiplicity needs to be specified. We're going to spend a good amount of time on that, but just note that here is where you capture that in the model.
   3. Leave the rest of the defaults alone.

ii. If you accidentally build the relationship from child to parent, you can just pick the opposite direction.

iii. Note that all relationships in UML are binary, that is, there are only two classes which can participate in the association.

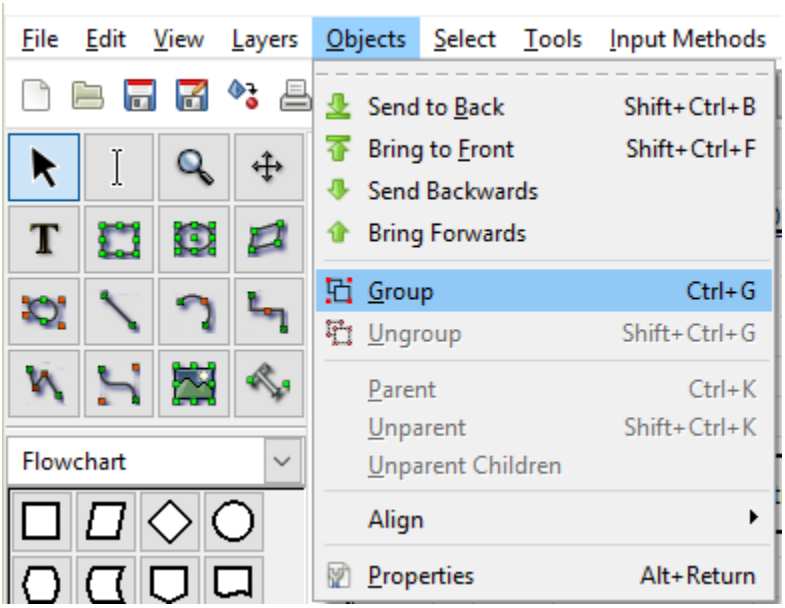
j. Quality checks
   i. Never forget the multiplicity of the associations between your classes.
   ii. Never leave the associations unlabeled.
   iii. Class names are singular, relation names are plural.
   iv. Remember, UML diagrams are conceptual:
      1. Never put migrated foreign keys into your UML diagrams.
      2. Never put surrogate keys into your UML diagrams.

3. For a **relation scheme** diagram:
   a. Each of the rectangles in the relation scheme diagram maps to a distinct column in the physical table, and generally (more on that later) is the implementation of an attribute of the corresponding UML class. Making an attribute in the relation scheme:
      i. The various stencils of course have different objects in them, and ironically it turns out that the **flowchart** box is going to be particularly useful for us. It is an object that accepts text, which we will use for the column names.
      ii. At the base of the tools menu, select the down arrow and pick "Flowchart".



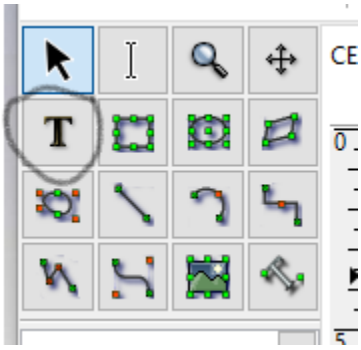
iii. Select the rectangular box, which formally is a Process/Auxiliary Operation, and position it where you need for it to go in your diagram.
iv. Start entering text. The box will automatically grow to accept the text. The default height of the box will work fine for us.

b. Multiple columns in your relation scheme.
   i. You will nearly always have multiple columns in your relation scheme, so create additional boxes for the other attributes as needed, get them positioned, and then group them into a single object.
      1. Be sure that you have the text (column names) that you need in each of the column boxes of your relation scheme.
      2. Optionally, to keep the attributes of a given relation scheme (each collection of columns joined together forms a relation scheme) together, you can group them in DIA.
         a. Select all the boxes of your relation scheme.
         b. Then group them into a single object.



c. If you need to change the text or move the boxes relative to each other, you will need to **un**group the boxes.

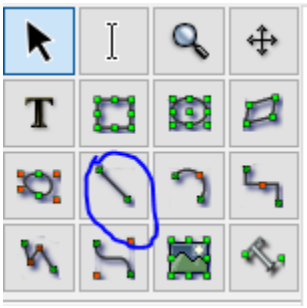
c. Labeling the relation scheme will be done with a text tool:
   i. Select the text object tool click on the canvas where you want to place the text.



   ii. Then just start typing to put the text on your canvas.

d. Designating the primary key
   i. Create a primary key rectangle directly above the columns of the primary key. Note this may have to be broken up if the members of your primary key are not adjacent one to the other.
   ii. Double click the primary key box to get into its properties dialog.
   iii. Make the fill color = #FFC900.

e. Drawing the relationship line from the parent to the child:
   i. Just use the line tool:



ii. It doesn't really matter which end is the parent, the diagramming tool imparts no real semantics to the relationship line in this context.
iii. Be sure to use attach points at both ends.

f. Designating foreign key
   i. There should be one foreign key shown on the diagram for each migrated foreign key. Remember that a given relationship can migrate multiple foreign keys.
   ii. If those keys are not adjacent to each other in the relation scheme, you will have to make a box "layer" for them.
   iii. Make the fill color = #1E8EFF to get about the right shade.

g. Cut and paste –
   i. Remember that the first box is the hardest. The first relation is the hardest. After that, it's all about cut and paste to make things easier.
   ii. You might want to take an entire relation in your model and group it once you're happy with the way that it looks. This makes it easier to move all of the pieces together.

h. Modeling standards:
   i. Every relation needs a primary key, regardless of whether that relation is a parent or not.
   ii. Draw the relationship line from the Primary Key of the parent to the migrated **Foreign** Key of the child.
   iii. Never forget the cardinality of the relationships.
   iv. Pay attention to the colors of the primary key and foreign key boxes. That helps make the model much easier to read.

4. Generally speaking
   a. Have the parent above the child.
   b. Be sure to label everything.

WHAT TO TURN IN:

- Your .dia model for the Customers/Orders tables. It should look as much like the example up above as possible.