CS 332/780: Object-Oriented Databases Fall 2015

PROJECT 2: Establishing Relations in the PC Model

Due: 11/15/15, Sunday, 11 PM

First, critically review your Project 1 model to detect flaws and shortcomings, including the comments made on the grade report, and remodel it to eliminate the problems. In the next stage, establish the following relations.

1. A relation storing which company makes which product.
2. Relations storing PCs' preinstalled hardware components, including components that can be preinstalled as option at the time of purchase.
3. Relations storing PCs' additional/replacement hardware components that can be installed after the purchase.
4. Relations storing which external (detached) hardware component can be connected to which PC (e.g., printer, keyboard, display monitor, mouse).
5. A plug-in compatibility relation of hardware components storing which component is replaceable by which component.
6. Relations storing PCs' preinstalled software components, including components that can be preinstalled as option at the time of purchase.
7. Relations storing which software product, as additional/replacement component, can be installed on (i.e., is supported by) which PC and/or OS platform.
8. Relations storing plug-in application software components. For example, a browser can work with a variety of file-viewing/processing software components, including text, document, image/movie, audio files.

You will have to make sound decisions on the choice of association, strong-composition, or aggregation relations to model the above relations.

You may have to make adjustments to the inheritance hierarchy and class attributes in the process of modeling the above relations. This kind of evolutionary remodeling is a natural and even necessary part of design and modeling.

Use the following checkpoints to help improvement and (re-)modeling.

- No relation names/multiplicities are missing, and all multiplicities are correct. Provide names/multiplicities for whole-part relations, not just for association relations.
- Inheritance and whole-part relations are not confused. Do not use inheritance for what should be a whole-part relation, and vice versa. The exception is the method of using strong composition to simulate inheritance as described in Course Notes #6.
- Whole-part relations are not used for what should be association relations. Do not use a whole-part relation unless it makes semantic sense to regard one class's objects as parts/components of the other class's objects.
- Aggregation (weak composition) is not used where strong composition should be used, and vice versa.

- Whole-part relations are pointing in the correct direction. Attach black/white diamonds to the whole classes not the part classes.
- An ancestor class's attributes are not redeclared in descendant classes.
- Common attributes are not distributed in subclasses. *Elevate* common attributes of descendant classes to as high an ancestor class as possible. Create parent/ancestor classes for this purpose when appropriate.
- Make classes *abstract* if you are certain they will not have any direct objects.
- Document the semantics (meaning) of classes and relations if their names are not self-explanatory.
- In UML class diagrams, do not use an attribute to represent a binary relation. That is, do not represent a relation R: A → B by an attribute of A containing a string or "ID" value of the related B-object. This may be appropriate in relational tables, but not in OO models.

Superimposing all the relations on the inheritance hierarchy would make the diagram clumsy. So I strongly recommend creating a separate "relation diagram" containing all the relations and their argument classes, without inheritance relations. This relation diagram will be shown under the inheritance-hierarchy diagram. In this process, use Visual Paradigm's copy-and-paste function for making copies of a class box, which will automatically copy all its features. Also, any change made to one copy of a class will be made to all copies automatically.

Finally, write a separate documentation concisely describing:

- the improvements made on your Project 1 model; and
- the relation schemas in your class diagram used to model the eight types of relations listed above. Make a numbered list of 1, …, 8, each listing the relation schemas in the format $R(C_1, \ldots, C_n)$.

Submission

Email to yukawa.qc@optimum.net your Visual Paradigm project file (the one with .vpp extension) and the documentation, using the subject header:

CS 332, Project 2, Your Full Name

The Visual Paradigm file must contain the improved inheritance hierarchy and the relation diagram.