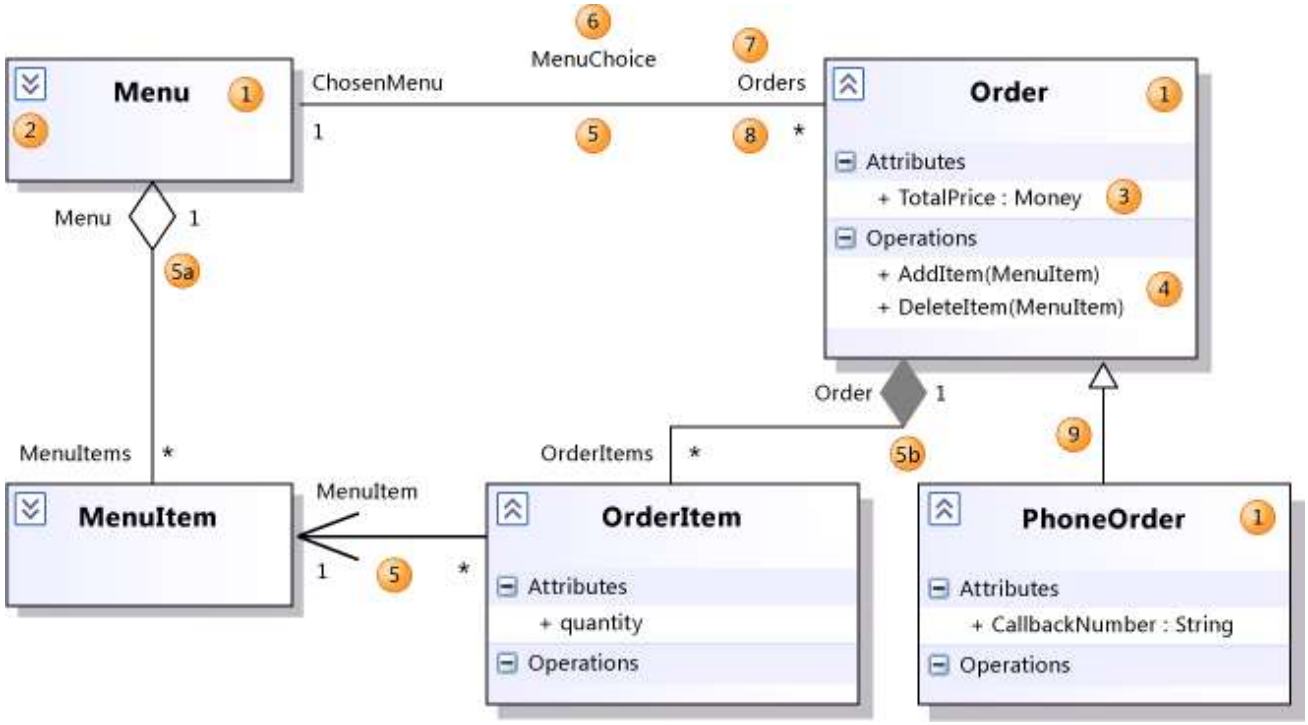
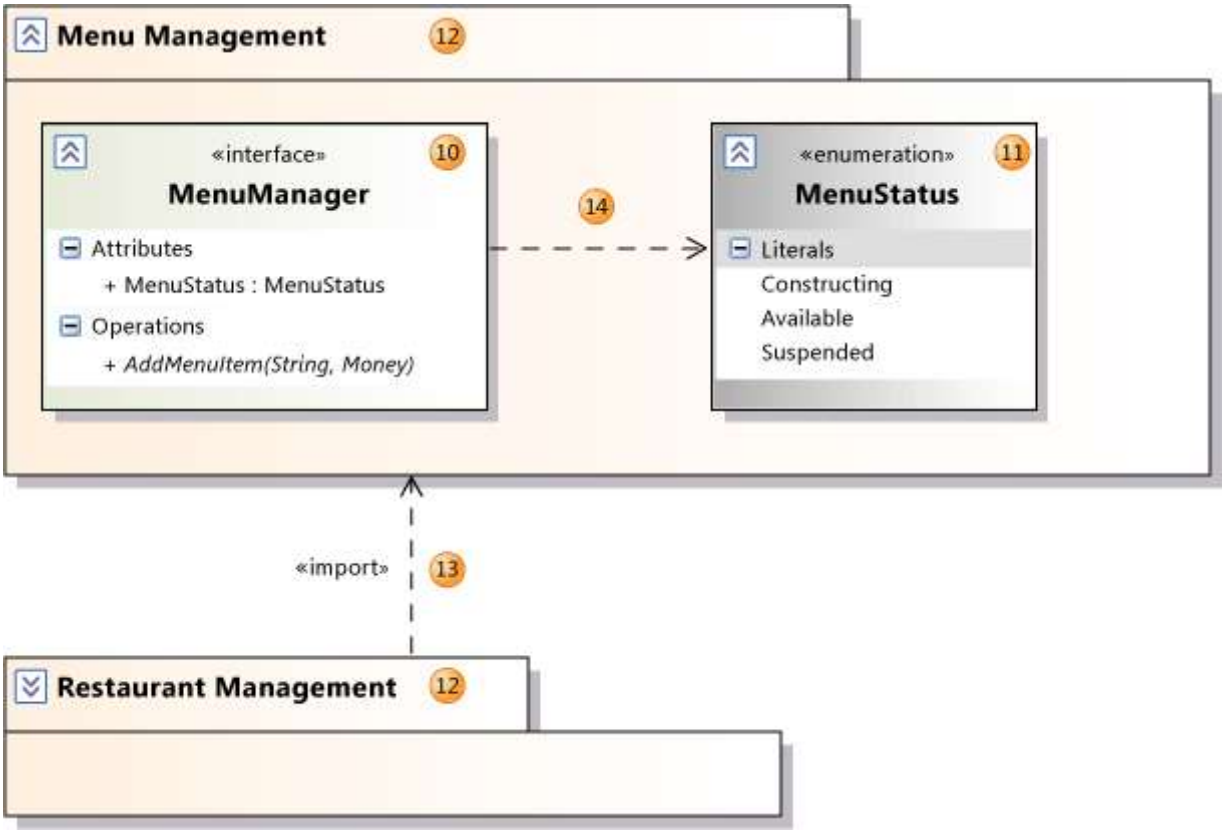


5: **Association**: A relationship between the members of two classifiers.

5a: **Aggregation**: An association representing a shared ownership relationship. The **Aggregation** property of the owner role is set to **Shared**.

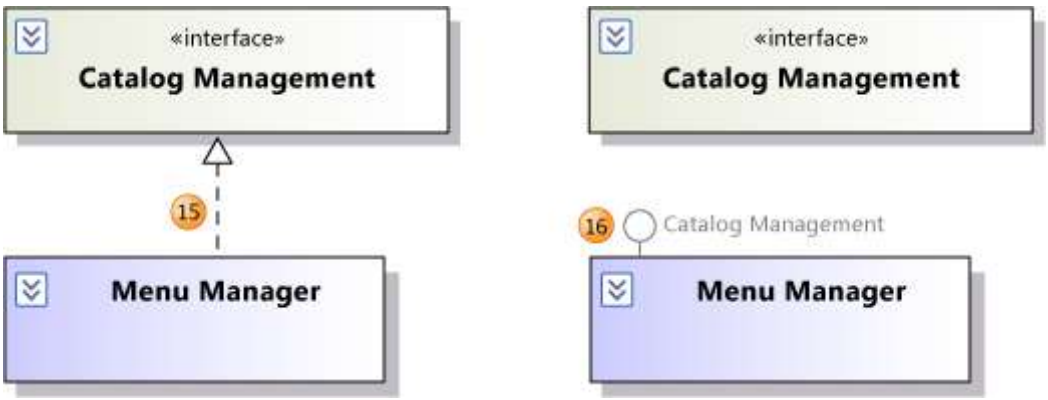
5b: **Composition**: An association representing a whole-part relationship. The **Aggregation** property of the owner role is set to **Composite**.

9: **Generalization**: The specific classifier inherits part of its definition from the general classifier. The general classifier is at the arrow end of the connector. Attributes, associations, and operations are inherited by the specific classifier. Use the **Inheritance** tool to create a generalization between two classifiers.



13: **Import**: A relationship between packages, indicating that one package includes all the definitions of another.

14: **Dependency**: The definition or implementation of the dependent classifier might change if the classifier at the arrowhead end is changed.



15: **Realization**: The class implements the operations and attributes defined by the interface. Use the **Inheritance** tool to create a realization between a class and an interface.

16: **Realization**: An alternative presentation of the same relationship. The label on the lollipop symbol identifies the interface.

UML Class Diagrams: Guidelines:
http://msdn.microsoft.com/library/dd409416%28VS.140%29.aspx

> **Properties of an Association**
>
> **Aggregation**: This appears as a diamond shape at one end of the connector. You can use it to indicate that instances at the aggregating role own or contain instances of the other.
>
> **Is Navigable**: If true for only one role, an arrow appears in the navigable direction. You can use this to indicate navigability of links and database relations in the software.
>
> **Generalization**: Generalization means that the specializing or derived type inherits attributes, operations, and associations of the general or base type. The general type appears at the arrowhead end of the relationship.
>
> **Realization**: Realization means that a class implements the attributes and operations specified by the interface. The interface is at the arrow end of the connector.

Let me know if you have more questions.

edited Sep 8 at 17:55

blkpingu
1,002 12 27

answered Feb 19 '10 at 2:42

Esther Fan - MSFT
7,778 4 24 25

1 Nice reference but to me a Menu -> MenuItem has the same relation like a Order -> OrderItem so both of them are Compositions. – Ignacio Soler Garcia Jun 28 '13 at 10:53

4 That both only mean, that the order item belongs to an order and can't be moved, whereas the Menu Item can be adjustable - the user may can to change the position of Menu Item. It is the solution chosen. Why not? – Gangnus Feb 3 '14 at 13:17

@Gangnus, thank you. That explanation clarified the difference that has eluded me for a long time. – JMD Jan 13 '15 at 18:06

1 @JMD, The order items can be moved as well. Composite aggregation is defined in the UML spec as follows: Composite aggregation is a strong form of aggregation that requires a part object be included in at most one composite object at a time. If a composite object is deleted, all of its partinstances that are objects are deleted with it. A part object may (where otherwise allowed) be removed from a composite object before the composite object is deleted, and thus not be deleted as part of the composite object. – www.admiraalit.nl Jan 26 '16 at 10:43

2 @aGer Thanks, I've updated the topic and image links. – Esther Fan - MSFT May 22 '17 at 16:39