PACKAGES

- a mechanism for organizing elements into groups
- package name must be unique within its enclosing package
- package must have a name that distinguishes it from other packages
- two naming mechanism simple name, path name(prefixed by the name of the package in which that package lives). Figure:1.
- package may contain classes, interfaces, components, nodes, collaborations, use cases, diagrams, and even other packages
- package forms a namespace(every element in a package can be identified uniquely eg:P1:: Queue and P2:: Queue are different)
- contents of a package can be shown textually or graphically. Figure 2.
- visibility : +, -, #
  1. Public elements (denoted by +) are visible outside the packages also.
  2. Protected elements (denoted by #) are visible only to packages that inherit from another package
  3. private elements (denoted by -) are not visible outside the package.
- fully qualified name example Client::OrderForm.
- two stereotypes import and access- both specify that the source package has access to the contents of the target. Import adds contents to source, so chances of name clashes, access doesn't add contents. These are non transitive

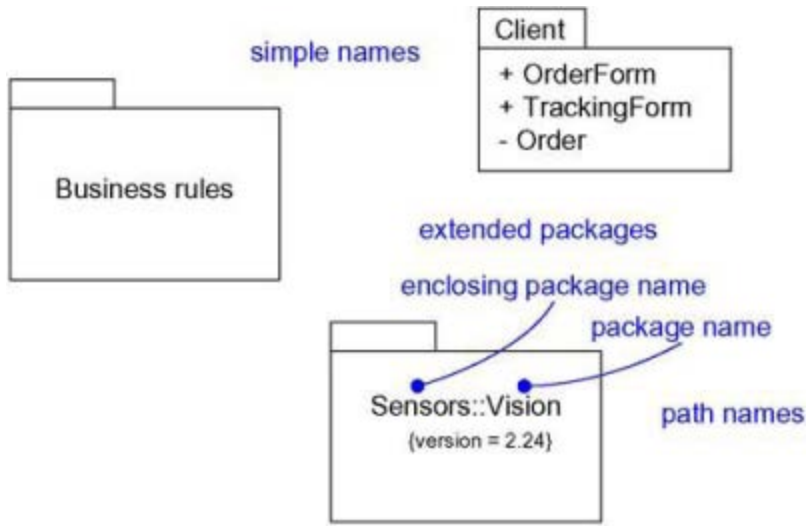

Figure1:Simple and Extended Package

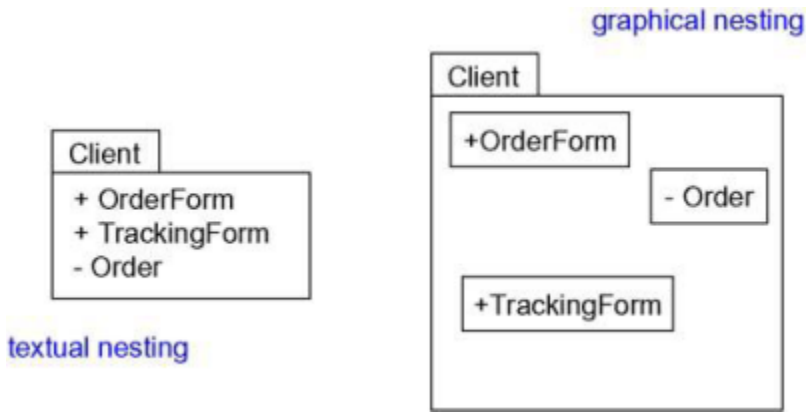

Figure2:package contents

Importing and Exporting

Importing means accessing the elements of the source by the target. It can be done by using the stereotype <<import>>. It's a one way process. It's non transitive. if A's package imports B's package, A can now see B, although B cannot see A. Importing grants a one-way permission for the elements in one package to access the elements in another package.

Exportsare the public parts of a package. It's also non-transitive Figure:3 represents this.

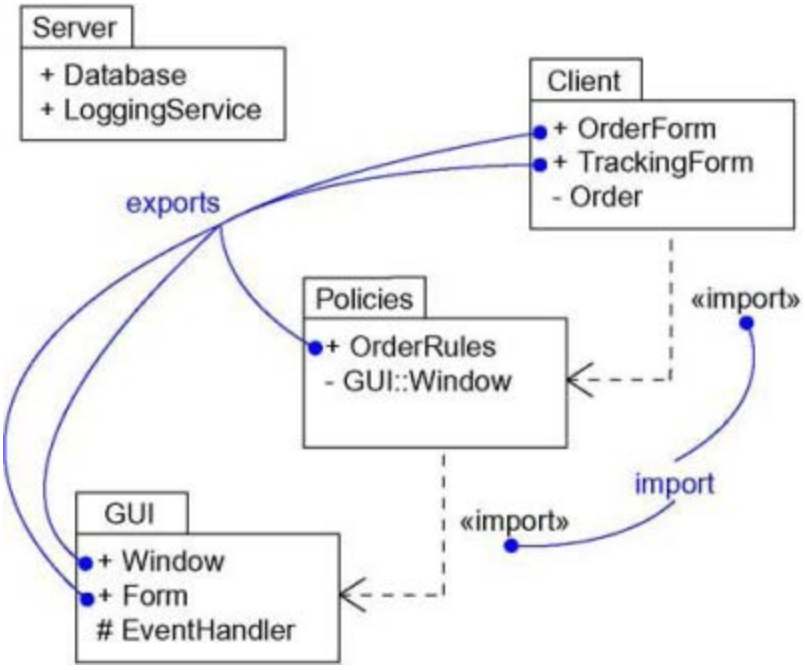

Figure:3 Importing and Exporting

Generalization

In Generalization the specialized packages inherit the public and protected elements of the more general package. Figure 4 illustrates this.

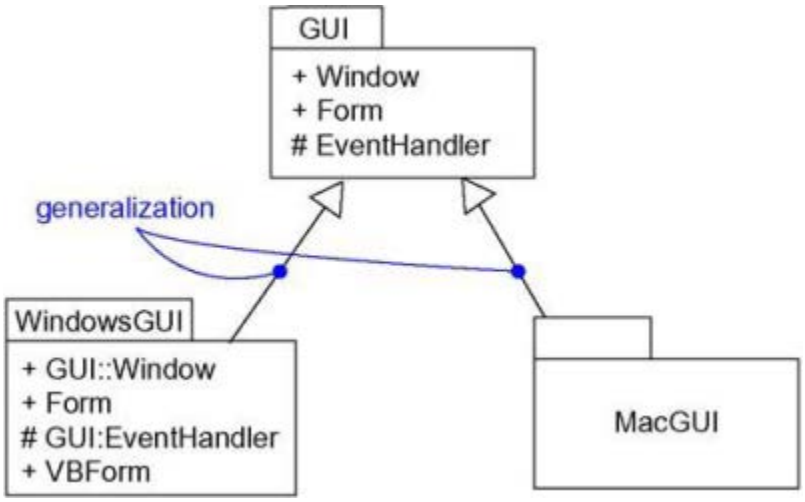

Figure:4 Generalization Among Packages

UML defines five standard stereotypes that apply to packages.

1. facade – Specifies a package that is only a view on some other package

2.framework – Specifies a package consisting mainly of patterns

3. stub – Specifies a package that serves as a proxy for the public contents of another package

4. subsystem – Specifies a package representing an independent part of the entire system being modeled

5. system – Specifies a package representing the entire system being modeled

Distinction between classes and packages:

Classes are abstractions of things found in a problem or solution; packages are mechanisms we use to organize the things in your model. Packages have no identity, classes do have identity through instances.

Modeling Groups of Elements

To model groups of elements,

- Scan the modeling elements in a particular architectural view and look for **clumps** defined by elements that are conceptually or semantically close to one another.
- Surround each of these clumps in a package.
- For each package, distinguish which elements should be accessible outside the package. Mark them public, and all others protected or private. When in doubt, hide the element.
- Explicitly connect packages that build on others via import dependencies.
- In the case of families of packages, connect specialized packages to their more general part via generalizations

Modeling Architectural Views

To model architectural views,

- Identify the set of architectural views that are significant in the context of your problem. In practice, this typically includes a design view, a process view, an implementation view, a deployment view, and a use case view.
- Place the elements (and diagrams) that are necessary and sufficient to visualize, specify, construct, and document the semantics of each view into the appropriate package.
- As necessary, further group these elements into their own packages.

- There will typically be dependencies across the elements in different views. So, in general, let each view at the top of a system be open to all others at that level.

Source : http://praveenthomasln.wordpress.com/2012/03/29/packages-s8-cs/