**What Is the UML?**

The Unified Modeling Language (UML) is a family of graphical notations,

backed by single meta-model, that help in describing and designing software

systems, particularly software systems built using the object-oriented (OO)

style.

Graphical modeling languages have been around in the software industry for

a long time. The fundamental driver behind them all is that programming languages

are not at a high enough level of abstraction to facilitate discussions

about design.

**Ways of Using the UML**

**UML as sketch**: emphasis is on selective communication rather than complete specification.

**UML as blueprint**: intend to be comprehensive, often with the aim of reducing programming to a simple and fairly mechanical activity.

**UML as programming language**: developers draw UML diagrams that are compiled directly to executable code, and the UML becomes the source code.

. Forward Engineering

. Reverse Engineering

. Conceptual Perspective

. Software Perspective

**Notations and Meta-Models**

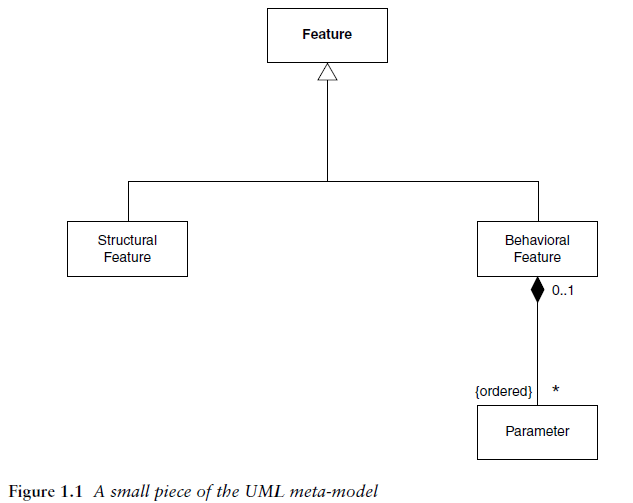
The **notation** is the graphical stuff you see in models;

it is the graphical syntax of themodeling language.

**meta-model:** a diagram, usually a class diagram, that defines the concepts of

the language.

Intuition rather than to formal definition.

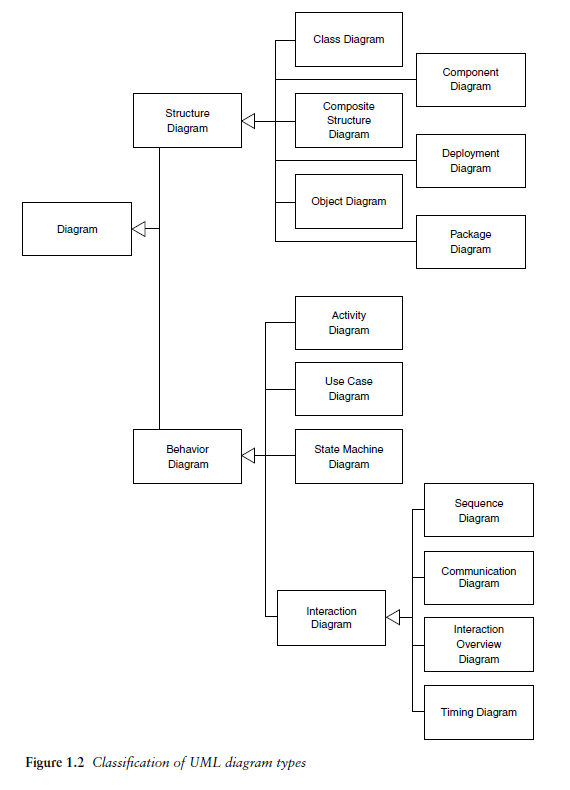


**TIP** - As you get deeper into the more detailed usage of the UML, you realize that

you need much more than the graphical notation. This is why UML tools are so

complex.

**UML Diagrams**



**What Is Legal UML?**

**prescriptive rules**

**descriptive rules**

**TIP** - You cannot look at a UML diagram and say *exactly* what

the equivalent code would look like. However, you can get a *rough idea* of what

the code would look like.

**UML Is Not Enough**

You shouldn’t hesitate to use a non-UML diagram if no UML diagram

suits your purpose.

. Screen flow diagram

. Decision table