<http://www.ibm.com/developerworks/rational/library/769.html>

The class diagram shows how the different entities (people, things, and data) relate to each other; in other words, it shows the static structures of the system. A class diagram can be used to display logical classes, which are typically the kinds of things the business people in an organization talk about.

The upper section shows the class's name; the middle section contains the class's attributes; and the lower section contains the class's operations (or "methods").

For inheritance relationship, a line with an arrowhead (complete rectangle) at the top pointing to the super class

Association relationship should be a solid line if both classes are aware of each other

A line with an open arrowhead if the association is known by only one of the classes.

Sequence diagrams show a detailed flow for a specific use case or even just part of a specific use case. They are almost self explanatory; they show the calls between the different objects in their sequence and can show, at a detailed level, different calls to different objects.

A sequence diagram has two dimensions: The vertical dimension shows the sequence of messages/calls in the time order that they occur; the horizontal dimension shows the object instances to which the messages are sent.

1-putting each class instance inside a box

2-put the class instance name and class name separated by a space/colon/space " : " (e.g., myReportGenerator : ReportGenerator

Important:If a class instance sends a message to another class instance, draw a line with an open arrowhead pointing to the receiving class instance; place the name of the message/method above the line.

statechart diagram

The statechart diagram models the different states that a class can be in and how that class transitions from state to state.

Only classes with "interesting" states - that is, classes with three or more potential states during system activity - should be modeled.

statechart diagram has five basic elements: the initial starting point, which is drawn using a solid circle; a transition between states, which is drawn using a line with an open arrowhead; a state, which is drawn using a rectangle with rounded corners; a decision point, which is drawn as an open circle; and one or more termination points, which are drawn using a circle with a solid circle inside it. To draw a statechart diagram, begin with a starting point and a transition line pointing to the initial state of the class. Draw the states themselves anywhere on the diagram, and then simply connect them using the state transition lines.

Activity diagrams

Activity diagrams show the procedural flow of control between two or more class objects while processing an activity. Activity diagrams can be used to model higher-level business process at the business unit level, or to model low-level internal class actions.

Component diagram

A component diagram provides a physical view of the system. Its purpose is to show the dependencies that the software has on the other software components (e.g., software libraries) in the system. The diagram can be shown at a very high level, with just the large-grain components, or it can be shown at the component package level.

The arrowed lined from component A to component B, means that component A is dependent on component B.

Deployment diagram

The deployment diagram shows how a system will be physically deployed in the hardware environment. Its purpose is to show where the different components of the system will physically run and how they will communicate with each other. Since the diagram models the physical runtime, a system's production staff will make considerable use of this diagram.

The notation in a deployment diagram includes the notation elements used in a component diagram, with a couple of additions, including the concept of a node. A node (three dimentional cube) represents either a physical machine or a virtual machine node (e.g., a mainframe node).