**Title: An Introduction to the Unified Modeling Language**

**Author: Laurie Williams 2004**

**Article Summary:**

This article demonstrates overview of Unified modeling language, its acceptance and standardization by Object Management Group. UML is a graphical or pictorial language which is an effective and easy step towards the software design. It uses some of Object Oriented Programming language concepts such as inheritance. Author has explained the importance of UML language and how UML with its different methods are used in every phase of software development life cycle and as software design process progress further. Usage of models and diagrams helps people to get better understanding of what is being built and leads to clarification early in the product development as author mentioned” a picture is worth a thousand words”. Also there is less scope for misunderstanding and users gets a better view of software in initial stages. Using UML to develop software design helps in causing less errors as pictures gives users broad interpretation of what their software would look like in future when it is ready to use. This research paper addresses the audience who are relatively new to UML and stepping towards learning basic UML concepts.

**Article Analysis:**

This research paper includes basic Unified Modeling language models which are used for software development. UML uses graphical representation rather than text to proceed with SDLC which in a way comforts users to get an idea of the product, as users are ignorant of technological terms but they know what they want from their product. Below is the analysis of models which author enlightened in the whole paper:

**1. Use Case Diagrams:**

Use Case Diagrams are the pictorial description of how actors uses the system in order to achieve some result which is useful to actor. Use cases are drawn in the very first stage of SDLC i.e. in software requirements gathering/ development stage when users specify their requirements (what kind of product, what functionalities they want). Use cases stands for functional requirements only. Use cases are symbolized by an ellipse i.e. oval shape with some words to define what that use case is doing. Generally it has verb which signifies some kind of action for e.g. Open outer door. An Actor is the one who interact with the system, gives some input to initiate the whole activity and receives desired output from system. Actor is represented by stick figure and generally have noun as its name. To show communication between the actor and the use cases, lines or arrows are drawn. Include and extend are the other two relationships which are used between use cases. Include is used when we have similar or common functionality being performed by actor. Extend is used for variations which are alternative flows.

**2. Class Diagrams**

Second Model explained by author in depth is Class Diagrams which can be drawn at high level and specific level created during requirement analysis and design phase of SDLC. The emphasis is on classes and their relationship in problem domain. The UML Class is represented by rectangular box with 3 partition in it. The top partition has class name on it. The middle one shows the class attributes and bottom one shows the class methods or operations. Class diagram incorporated in design phase has implementation level details in it. Then author discussed various relationships which are used in process of making Class diagrams as follows:

1. **Static Relationship**: Inheritance has specialization and generalization concept represented by an empty arrow pointing towards superclass from subclass. Association relationship between instances of class is denoted by line. Aggregation and composition are specialized types of association. Aggregation can be defined as the independent existence of part class and whole class and are unidirectional. Vice versa Composition is defined as dependent or compulsory for existence.
2. **Attributes & Operations:** Names of attributes are specified with visibility denoted by symbols as i. public with + sign ii. private with – sign iii. protected with # symbol. Default value is public. Operations or methods are the main responsibility or function for the class to operate upon. They are specified with return type, signature and visibility.
3. **Cardinality:** It shows relationships between objects of classes with numbers on it. The numbers are put on the association line to indicate the multiplicity. There can be one to one or one to many or zero to many or zero to one relationships.
4. System is divided into sub systems known as **packages**.
5. **Stereotypes:** They are used to give some additional meaning to the information being shown to model elements. This information is written between guillemots (<< >>).Also we can define our own stereotypes.
6. **Object Diagrams:** Designed to reveal point in time detailed state of system with many objects of the same class. It has 2 compartments, top compartment has object name separated by colon and followed by class with underline & second has instance variables with values assigned. Also the links are used to show communication between the various objects. Author has explained object diagram with an example of monopoly gameboard.

**3. Sequence Diagrams**

As the name says sequence diagrams represents sequence of interaction between objects to pass messages using time as a variable. They are used in analysis and design phase of SDLC. Objects are drawn in rectangular box at the top of diagram and object name is written after colon and it is underlined. There are 2 dimensions of Sequence diagrams: Horizontal displays all the participating objects and vertical shows the Time. Interaction starts when message is sent from one object to another from top left corner. Communication is shown by arrows with some argument written over it.

**4. State Diagram**

Author then highlights on State Diagrams used during analysis & design phase with an example of turn in monopoly game. They are used for analyzing behavior of use cases and shows the states of the object at point of time throughout the life cycle. Whenever objects changes it state it is termed as transition. Transition occurs when the object is triggered by external event. Author says sometimes transitions has to pass through guard condition then if that permits then only it can change state.

**5. Activity Diagram**

Activity diagrams are advanced version of traditional flow charts as they can handle concurrent activities. Activity diagrams are used to realize use cases in design phase of SDLC. They show the up to date sequence of activities from beginning to end, they have entry and exit conditions. Activity diagrams leads to swimlanes diagrams which shows who does what actions. Writer gave an example of preparing corn to show activity diagram.

**Article Critique:**

This article gives insights for a beginner learning UML and idea about different diagram used at different stages of SDLC. Author has got a clear concept of UML which can be seen in text and description of paper. Based on my study here are the strengths and weakness:

**Strengths:**

* Information author trying to convey is totally acceptable and clear. Well defined terms and usage of example helps in understanding concepts clearly.
* It is clear which model to use when and at what stage of SDLC.
* Article was true to its name i.e. Introduction to the Unified Modeling Language. Beginners can get idea of what UML is all about.

**Weakness:**

* This paper did not include on how to write narrative use cases which is of significant importance in UML. Author must need to include that in Use case diagrams section.
* There is no new contribution made to UML or software industry with this research paper.
* Author has used the already established and developed knowledge to explain concepts and models of UML.
* The title will not catch attention of existing learners of the UML.