Conceptual Model of the UML

1. **The UML’s basic building blocks**
2. **The rules that dictate how the building blocks may be put together**
3. **Some common mechanisms**

Building Blocks of UML- **Things, Relationship and Diagrams**

1. **Things – Structural Things, Behavioral Things, Grouping Things and Annotational Things**
   1. **Structural Things / Classifiers**

**Nouns of the UML models.**

**Static part of model, representing elements that are either conceptual or physical.**

1. **Class**

**Set of objects sharing same attributes, operations, relationship and semantics**

1. **Interface**

**Collection of operations that specify a service of a class or component**

IPaint

Window

Window

IWindow

Window

1. **Collaboration**
2. **Use Case**
3. **Active Class**

|  |  |  |
| --- | --- | --- |
|  | **Event Manager** |  |
|  |
| **Operation1()**  **Operation2()** |

1. **Component**
2. **Artifacts**

|  |
| --- |
| **<<artifact>>**  **Window.dll** |

1. **Nodes**

Server

* 1. **Behavioral Things**

**Verbs of a model**

* + 1. **Interaction**

display

* + 1. **State machine**

Waiting

* + 1. **Activity**

Waiting

* 1. **Grouping Things**
     1. **Package**

Business Rule Package

* 1. **Annotational Things**

Return copy to Self

1. **Relationship**
2. **Dependency**
3. **Association**

\*

0..1

**Employer Employee**

1. **Generalization**
2. **Realization**
3. **Diagrams**

**Rules of UML**

* **Names**
* **Scope**
* **Visibility**
* **Integrity**
* **Execution**

Common Mechanisms in the UML

1. **Specifications**
2. **Adornments**
3. **Common Division**
4. **Extensibility Mechanisms**
5. Stereotypes
6. Tagged Values
7. Constraints

|  |
| --- |
| <<authored>>  EventQueue |
|  |
| Add()  Remove()  Flush() |

|  |
| --- |
| <<authored>>  Version=3.2  Author =cmp |

**Visibility**

public (+), private(-), protected(#), package(~)

**Attributes**

[*visibility*]name[':' type]['['multiplicity] ']']['=' initial-value][property-string {',', property string}]

+origin: Point Name

name: String[0...1] Name, Type and Multiplicity

origin: Point={0, 0} Name, Type and Initial Value

id: Integer {readonly} Name and Property

**Operations**

[visibility]name['{' parameter-list ')'] [':' return-type] [property-string{',' property-string}]

Display Name

+display Visibility and Name

set(n: Name, s: String) Name and Parameters

getID(): Integer Name and Return Type

restart() {gaurded} Name and Property

query, sequential, gaurded, concurrent, static

Generalization

1. complete

2. incomplete

3. disjoint

4. overlapping

Dependancies

* 1. bind – Specifies source instantiate the target template using the given actual parameter
  2. derive – Specifies that the source may be computed from the target
  3. permit – Specifies that the source is given special visibility into the target,