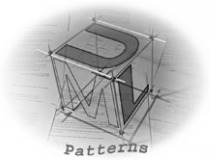


UML Extension Mechanisms

By Oscar Kozlowski

Presentation Outline

- What is UML
- How does UML help
- How is UML limited
- Introducing the UML Extension Mechanisms:
  - Stereotypes
  - Constraints
  - Tagged Values
- Example usage
- Conclusions

What is UML?

- Unified Modelling Language
- Standard language for modelling system blueprints
  - Software and non-software systems
  - Graphical notation to define the objects
  - Textual notation to fully define relationships
- Object-oriented and component based methodologies are fully integrated
  - Data encapsulation
  - Reusability
  - Extensibility

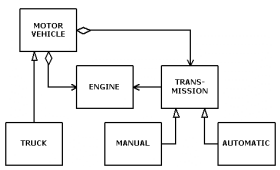
Ideology of UML

- UML is not tied to any specific development methodology or lifecycle
- Designed to incorporate current best software engineering practises but not force them
- Designed to integrate with CASE tools
- Ultimately a communication tool

How does UML help?

- UML helps visualise and document models of systems and processes in a manner that meets the requirements specification
- Helps the stakeholders visualise the product and the options available to them
- UML helps to visualise the important aspects of a system while abstracting the remainder away
- Allows developers to quickly assemble products from existing components and operations
- Helps explore the problem domain to assist risk management

UML Example



Limitations

- UML focuses on defining notation for the majority of users' modelling needs, but it cannot express every aspect of every model across every problem domain
- UML cannot express non-semantic attributes of a model effectively
- UML is a communication tool so user tailoring of the language that improves this communication is vital
- This tailoring must be done in a controlled manner

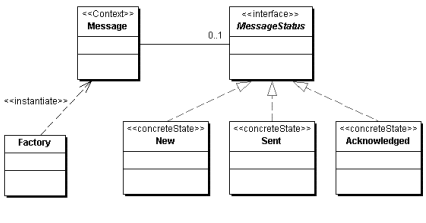
Solution

- These limitations are overcome with three defined extension mechanisms that enable further information about the system to be communicated
  - Stereotypes
  - Constraints
  - Tagged Values
- Powerful as semantically user defined
- Controllable as syntactically specification defined
- Used to define problem domain specific or implementation language specific extended information

Stereotypes

- A stereotype represents a variation of an existing model element with the same form (e.g. attributes or relationships) but with a different intent
- By default, stereotypes are depicted as keywords surrounded by guillemot characters « keyword »
- Until UML 1.4 (September 2001) was ratified, each diagram element could only have one stereotype. This has been lifted but not all CASE tools have caught up
- There are some pre-defined stereotypes in UML such as « use » and « extends »

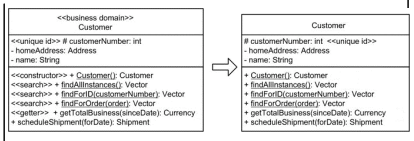
Stereotypes Example



Graphical Stereotypes

- Stereotypes can also be depicted graphically
- This can come at the expense of attributes and operations unless a graphical / textual hybrid is used
- Symbols used must be familiar and self explanatory

Going overboard

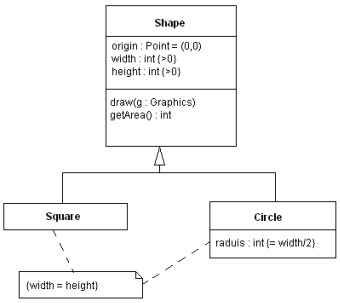


- List stereotypes after the attribute
- Don't indicate assumed stereotypes
- Prefer self explanatory naming schemes over stereotypes

Constraints

- Constraints specify some condition or rule about the modeling element that *must* be maintained as true
- Constraints has semantic impact
- There are some pre-defined constraints in UML
- Any constraint attached to a stereotype applies to every model element that has that stereotype
- Constraints are depicted after the model element / stereotype as text within curly braces { >0 }

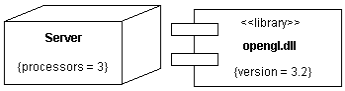
Constraints Example



Tagged Values

- A Tagged Value is a name-value pair denoting a property of a model element
- Tagged values also have semantic impact
- Not equivalent to class attribute as does not apply to an instance of the modeled object, but to the model element itself
- Used mainly as metadata or to specify properties relevant to the build environment
- Tagged values are depicted after the model element / stereotype as text within curly braces { author = Oscar}

Tagged Values Example



Conclusion

- UML modeling is all about communication
- Use UML extension mechanisms to convey specific additional structural and semantic information
- Be aware of not overdoing the use of UML extension mechanisms just for the sake of it