



OO Analysis and Design with UML 2 and UP

Dr. Jim Arlow,
Zuhlke Engineering Limited



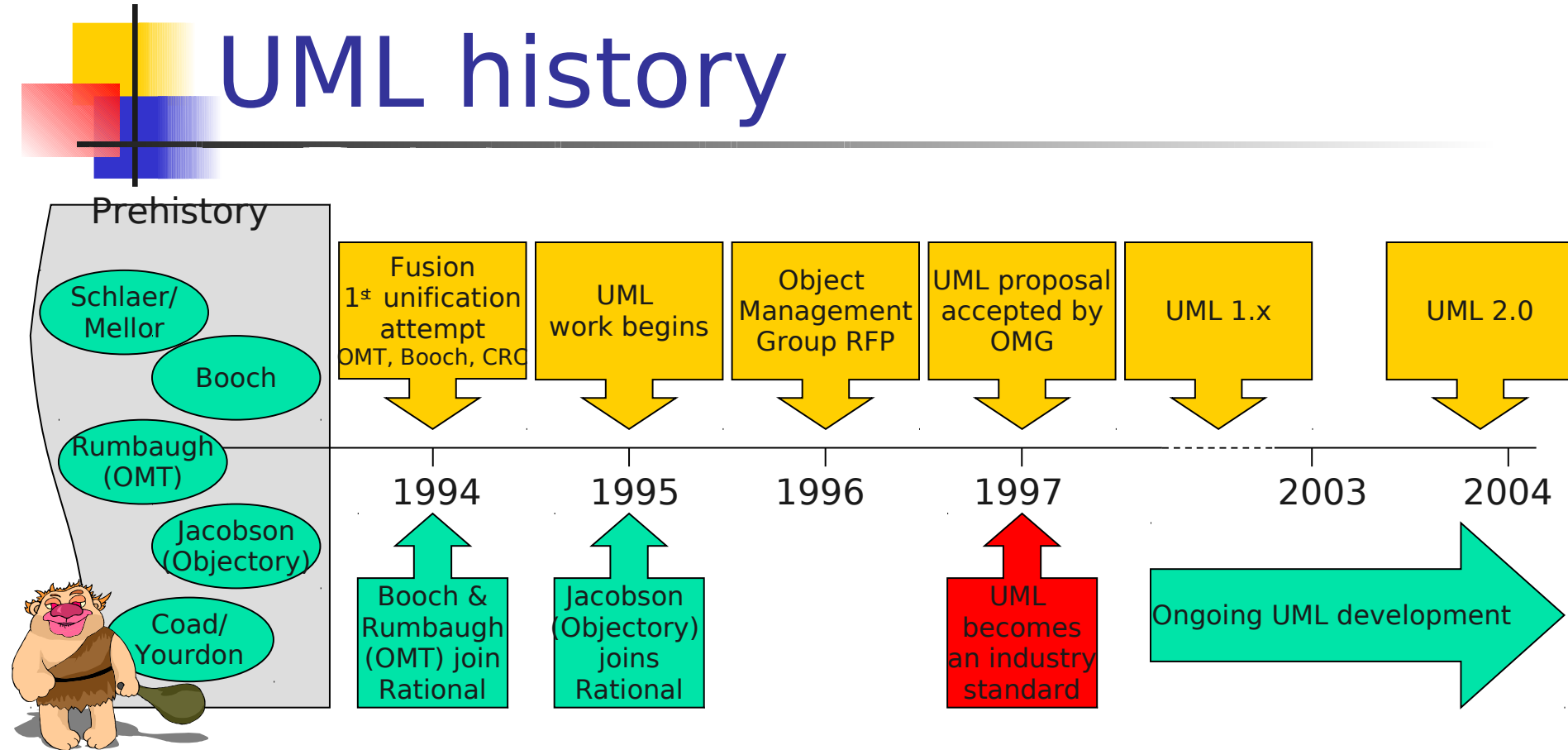
UML principles



What is UML?

- Unified Modelling Language (UML) is a general purpose visual modelling language
 - Can support all existing lifecycles
 - Intended to be supported by CASE tools
- Unifies past modelling techniques and experience
- Incorporates current best practice in software engineering
- UML is *not* a methodology!
 - UML is a visual language
 - **UP is a methodology**

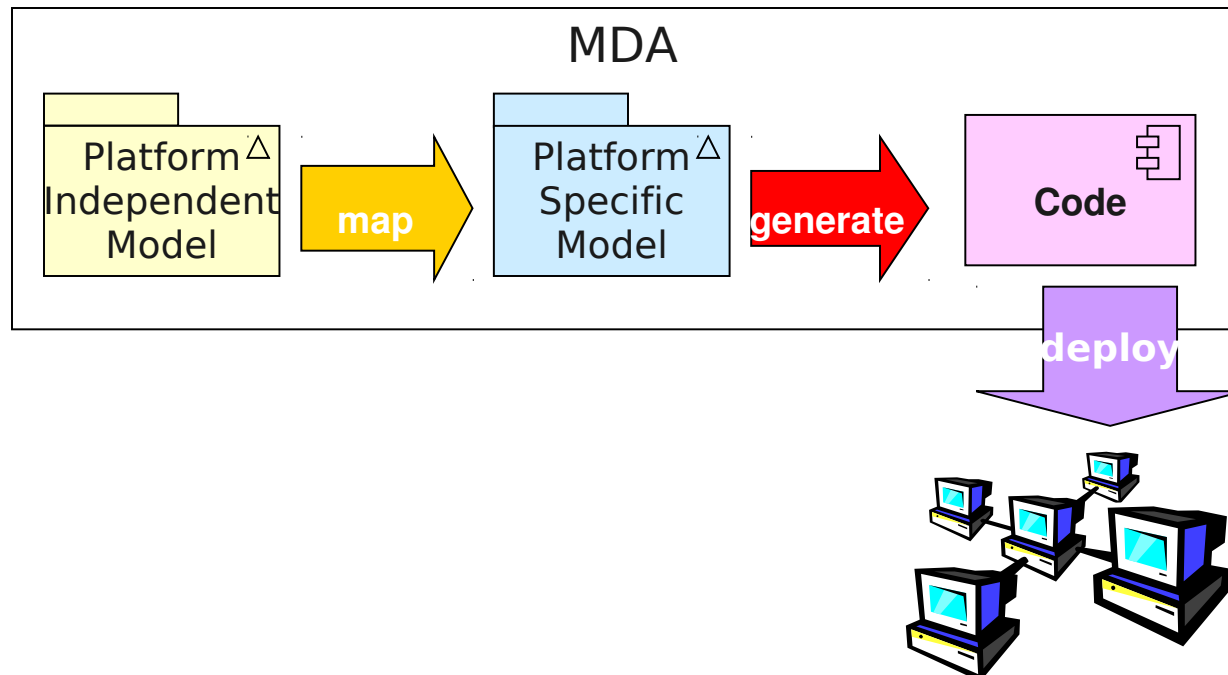
UML history



- A major upgrade to UML at the end of 2003:
 - Greater consistency
 - More precisely defined semantics
 - New diagram types
 - Backwards compatible

UML future?

- The future of development of UML will be increasingly affected by Model Driven Architecture (MDA)





Why "unified"?

- UML is unified across several domains:
 - Across historical methods and notations
 - Across the development lifecycle
 - Across application domains
 - Across implementation languages and platforms
 - Across development processes
 - Across internal concepts

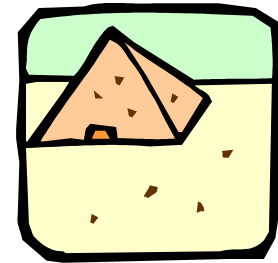
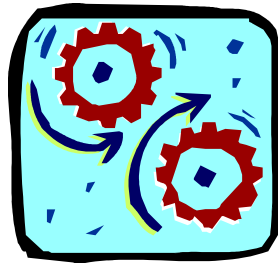
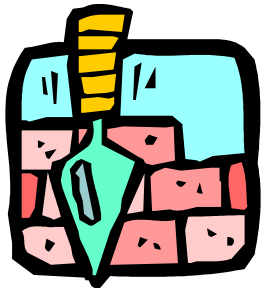


Objects and the UML

- UML models systems as collections of objects that interact to deliver benefit to outside users
- Static structure
 - What kinds of objects are important
 - What are their relationships
- Dynamic behaviour
 - Lifecycles of objects
 - Object interactions to achieve goals

UML Structure

- In this section we present an overview of the structure of the UML
- All the modelling elements mentioned here are discussed later, and in much more detail!

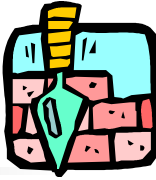


Building blocks

Common mechanisms

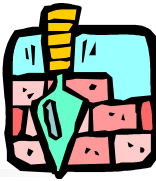
Architecture

UML building blocks

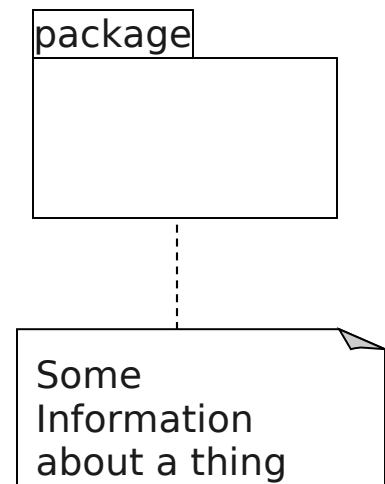


- Things
 - Modelling elements
- Relationships
 - Tie things together
- Diagrams
 - Views showing interesting collections of things
 - Are views of the model

Things

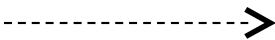

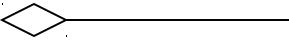
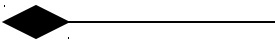
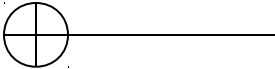
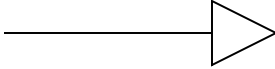
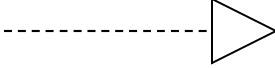


- Structural things – nouns of a UML model
 - Class, interface, collaboration, use case, active class, component, node
- Behavioural things – verbs of a UML model
 - Interactions, state machine
- Grouping things
 - Package
 - Models, frameworks, subsystems
- Annotational things
 - Notes
 - Tagged values



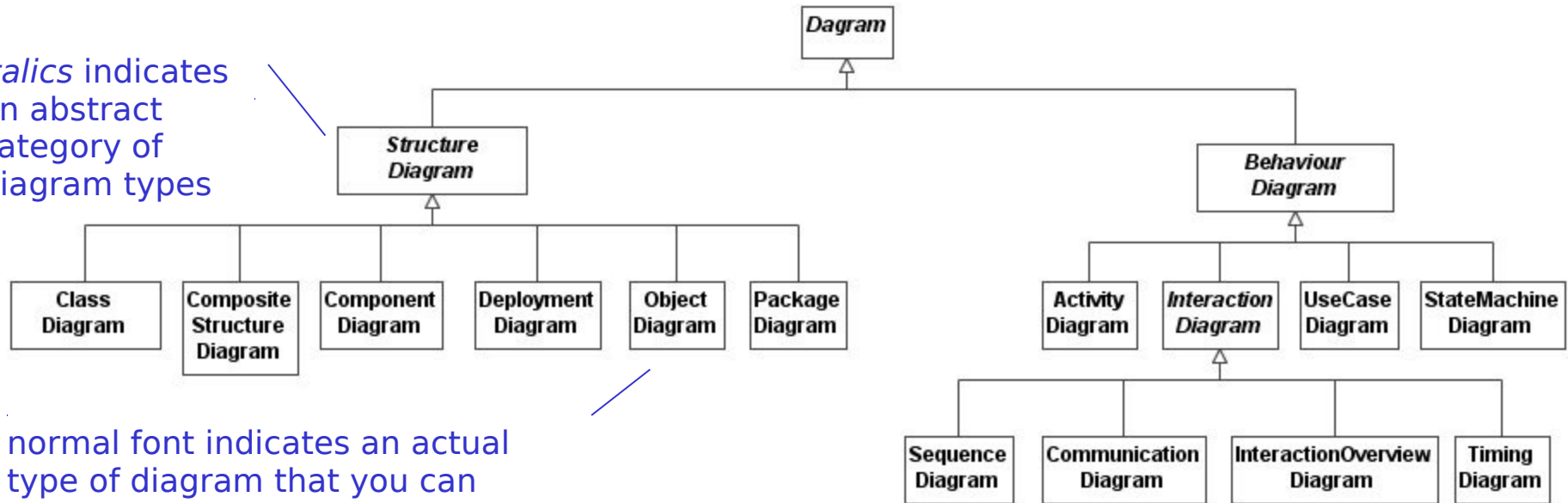
Relationships



relationship	UML syntax	brief semantics
dependency		The source element depends on the target element and may be affected by changes to it.
association		The description of a set of links between objects.
aggregation		The target element is a part of the source element.
composition		A strong (more constrained) form of aggregation.
containment		The source element contains the target element.
generalization		The source element is a specialization of the more general target element and may be substituted for it.
realization		The source element guarantees to carry out the contract specified by the target element

UML has 13 types of diagram

italics indicates an abstract category of diagram types



normal font indicates an actual type of diagram that you can create

- Structure diagrams model the structure of the system (the static model)
- Behavior diagrams model the dynamic behavior of the system (the dynamic model)
- Each type of diagram gives a different type of view of the model

UML 2 diagram syntax

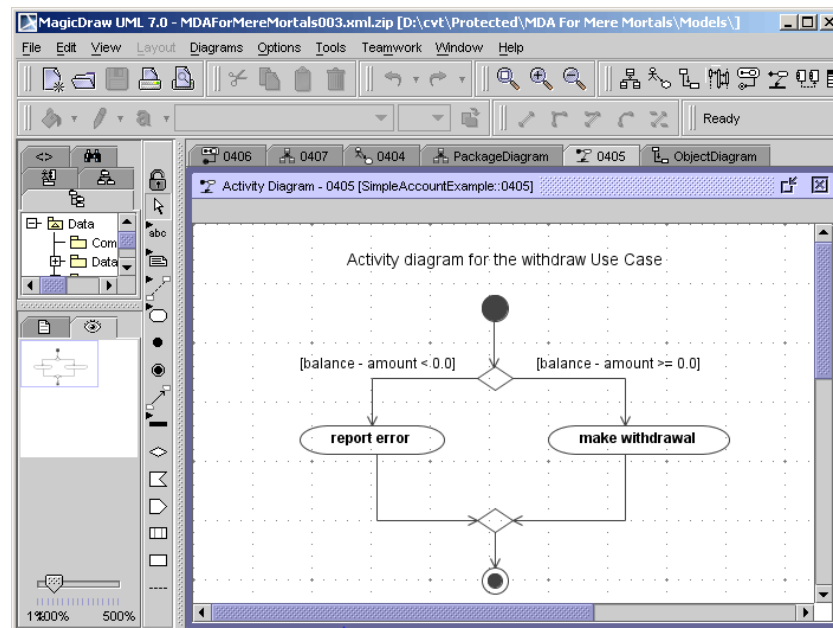


frame



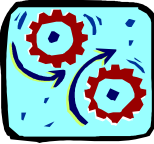
heading syntax: `<kind> <name> <parameters>`
 N.B. `<kind>` and `<parameters>` are optional

- The heading specifies the **kind** of diagram, its **name** and any information (**parameters**) needed by elements in the diagram
- The frame may be implied by a diagram area in the UML tool



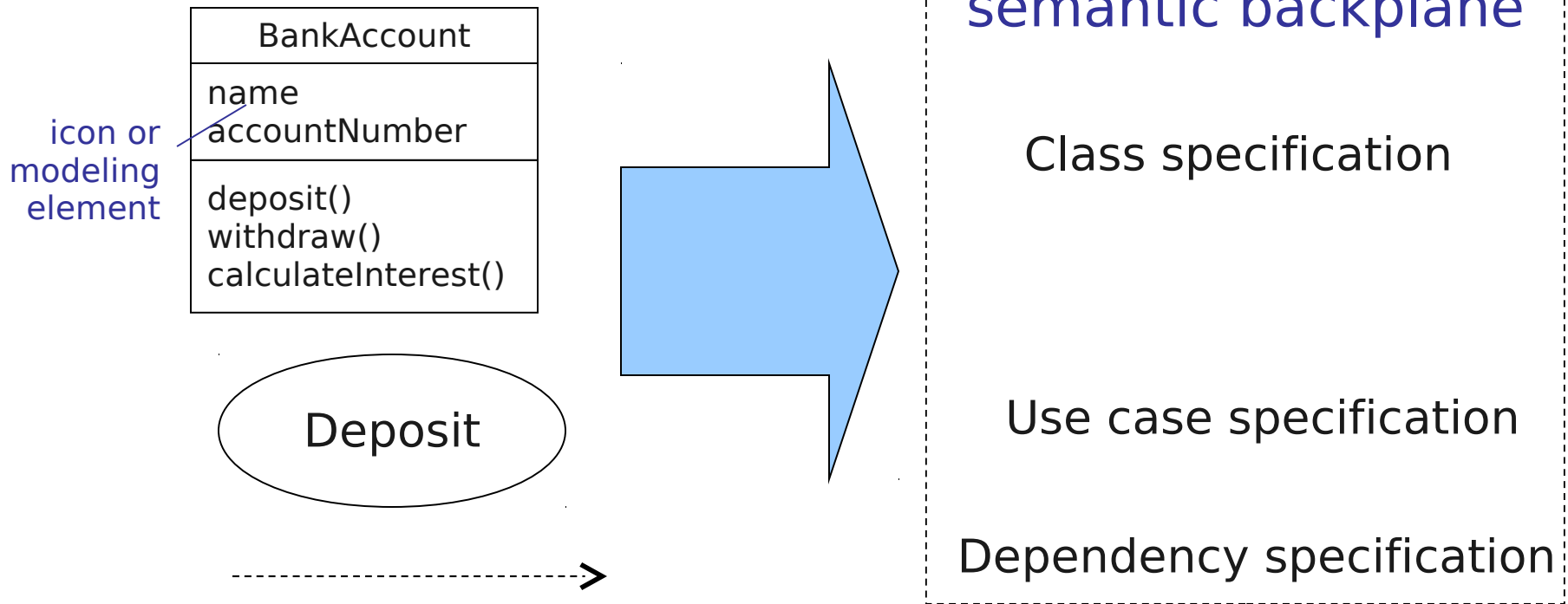
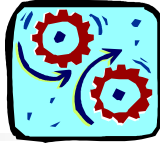
implied frame

UML common mechanisms



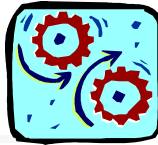
- UML has four common mechanisms that apply consistently throughout the language:
 - Specifications
 - Adornments
 - Common divisions
 - Extensibility mechanisms

Specifications

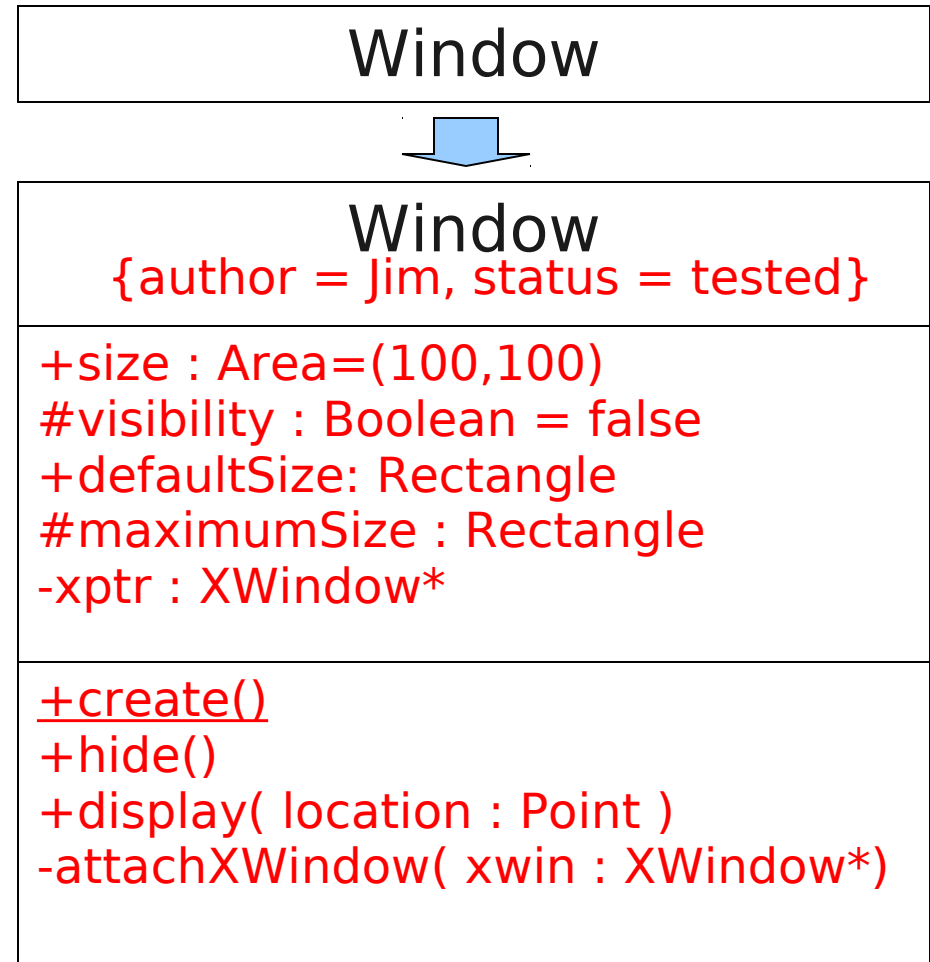


- Behind every UML modelling element is a *specification* which provides a textual statement of the syntax and semantics of that element
- These specifications form the *semantic backplane* of the model

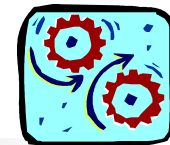
Adornments



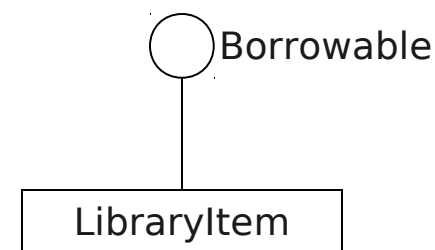
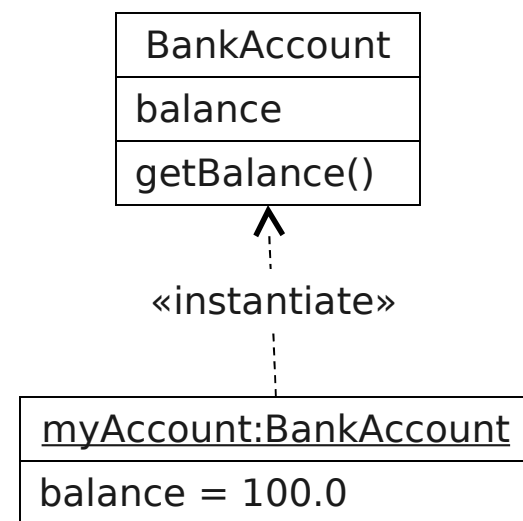
- Every UML modelling element starts with a basic symbol to which can be added a number of *adornments* specific to that symbol
- We only show adornments to *increase the clarity* of the diagram or to highlight a specific feature of the model



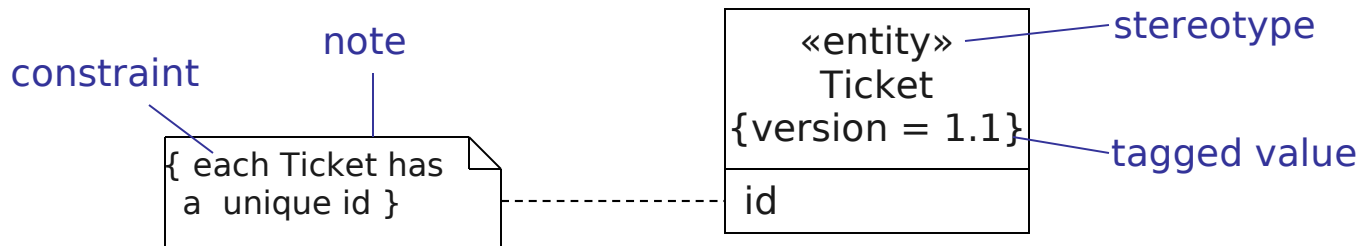
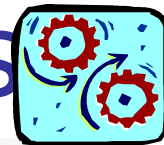
Common divisions



- Classifier and instance
 - A classifier is an abstraction, an instance is a concrete manifestation of that abstraction
 - The most common form is class/object e.g. a classifier might be a BankAccount class, and an instance might be an object representing my bank account
 - Generally instances have the same notation as classes, but the instance name is underlined
 - 33 types of classifiers
- Interface and implementation
 - An interface declares a contract and an implementation represents a concrete realization of that contract



Extensibility mechanisms



■ Stereotypes

- A stereotype allows us to define a new UML modelling element based on an existing one
- We define the semantics of the stereotype ourselves
- Stereotypes add new elements to the UML metamodel
- Written as «stereotypeName»

■ Constraints

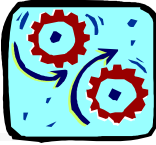
- Extends the semantics of an element by allowing us to add new rules about the element
- Written as { some constraint }

■ Tagged values

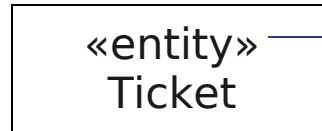
- Allows us to add new, ad-hoc information to an element's specification
- Written as { tag1 = value1, tag2 = value2 ... }

are attached to a stereotype

Stereotype syntax options



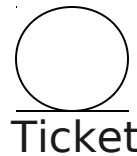
stereotype name
in guillemets



stereotype

preferred

stereotype icon



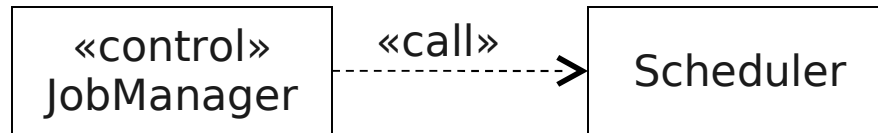
icon

preferred

stereotype name
and icon

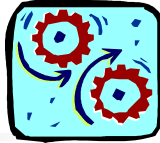


stereotyped
relationship



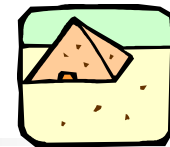
- A stereotype introduces a *new* modelling element and so we must *always* define semantics for our stereotypes
- Each model element can have many stereotypes

UML profiles

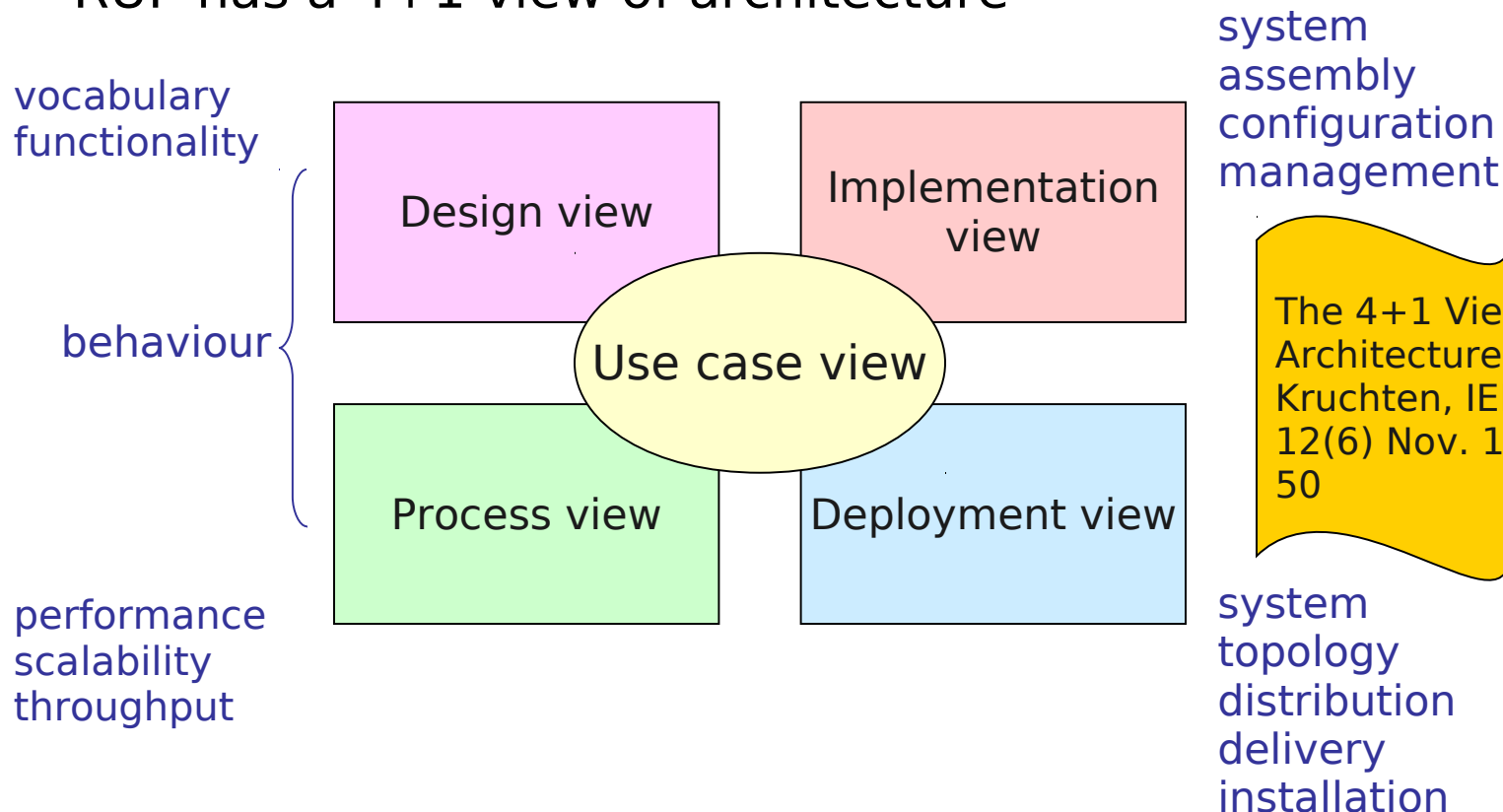


- A profile customizes UML for a specific purposes
- A UML profile is a collection of stereotypes, tagged values and constraints
 - The tagged values and constraints are associated with stereotypes
- Stereotypes extend one of the UML meta-model elements (e.g. Class, Association)
 - Any element that gets the stereotype also gets the associated tagged values and constraints

Architecture



- "The organisational structure of a software system"
 - UML specification & IEEE Std. 610.12-1990
 - RUP has a 4+1 view of architecture

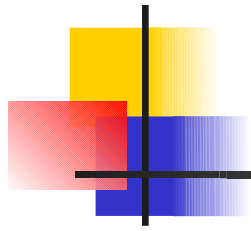


The 4+1 View of Architecture, Philippe Kruchten, IEEE Software, 12(6) Nov. 1995, p. 45-50



Summary

- The UML is composed of *building blocks*:
 - Things
 - Relationships
 - Diagrams
- The UML has four *common mechanisms*:
 - Specifications
 - Adornments
 - Common divisions
 - Extensibility mechanisms
- The UML is based on a 4+1 view of *system architecture*



Thanks for your attention