

## UML

### The Unified Modeling Language

*What is it and how can it help  
the Object-Oriented SAS® developer ?*



Willy Waks  
2W Systems Co., Inc.  
Dallas, Texas  
Tel: (972) 661-8619  
Cellular: (972) 849-9666  
Email: wwaks@swbell.net

8/31/99

1

2W Systems Co., Inc.

## Credits

- Parts of this presentation are excerpts from materials which are:

- ◆ Copyright © Rational Software Corporation.
- ◆ Copyright © Martin Fowler, *UML distilled*, Addison-Wesley, 1997
- ◆ Copyright © SAS Institute.

8/31/99

2

2W Systems Co., Inc.

## Introduction

- OOAD: Object Oriented Analysis and Design.
- UML: Unified Modeling Language. It is now the standard language used to specify, visualize and document the artifacts of an object oriented system, as well as for business modeling and other non-software systems.

8/31/99

3

2W Systems Co., Inc.

## Scope

- Provide a bird's eye view of OOAD
- Provide a bird's eye view of UML
- Relate the concepts to the world of SAS environment
- Encourage you to enrich your expertise in SAS with knowledge of these technologies

8/31/99

4

2W Systems Co., Inc.

## Traditional A&D Methodologies

- Structured Analysis and Design Development (J.L. Warnier, Tom De Marco, Yourdon)
- Relational Modeling (E.F Codd, C.J. Date)
- BPR: Business Process Reengineering
- Flow charts and Diagramming Techniques (James Martin)
- ERD (Entity Relationship Diagrams), DFD (Data Flow Diagrams)
- These techniques are still valuable

8/31/99

5

2W Systems Co., Inc.

## OO Technology on mainframes

- Common misconception: Object Oriented technology is just for flashy graphical interfaces
- You can design OO systems for mainframes, even COBOL !  
For ex. visit <http://extra.newsguy.com/~oocobol/>
- Why ? Because Components can be:
  - ◆ non-visual: ( MODELS )
  - ◆ or visual ( CONTROL )

8/31/99

6

2W Systems Co., Inc.

## OO Technology on mainframes

- The key word is METADATA !, i.e Data on Data
- Note: Most of the build-time enhancements for SAS/AF software require the graphical capabilities of desktop environments and, as such, will not work in mainframe environments. However, applications that are developed with SAS/AF in other operating environments will execute on mainframe platform

8/31/99

7

2W Systems Co., Inc.

## UML

- The UML provides improved expressiveness and holistic integrity.
- The UML is a (mainly graphical ) notation used to express Object Oriented designs.
- UML unifies the methods of Booch, Rumbaugh (OMT) and Jacobson which appeared in the late '80s and early '90s
- Standardized by the OMG (Object Management Group)

8/31/99

8

2W Systems Co., Inc.

## UML

- Objectory : the process for OO development (like IBM life-cycle) is under standardization
- You don't need to use Objectory to use UML
- OOAD and Objectory constitute together the concept of *Information Engineering*

8/31/99

9

2W Systems Co., Inc.

## UML Conceptual Framework

- The generally accepted conceptual framework for metamodeling is based on an architecture with four layers:
  - ◆ meta-metamodel
  - ◆ metamodel
  - ◆ model
  - ◆ user objects

8/31/99

10

2W Systems Co., Inc.

## UML Conceptual Framework

| Layer                    | Description                                                                                        | Example                                                                                      |
|--------------------------|----------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|
| meta-metamodel           | The infrastructure for a metamodeling sublanguage. Defines the language for specifying metamodels. | MetaClass, MetaAttribute, MetaOperation                                                      |
| metamodel                | An instance of a meta-metamodel. Defines the language for specifying a model.                      | Class, Attribute, Operation, Composite                                                       |
| model                    | An instance of a metamodel. Defines a language to describe an information domain.                  | SearchForm, addPrice, sellLimitOrder, SearchQuoteServer                                      |
| user objects (user data) | An instance of a model. Defines a specific information domain.                                     | <AccountSoftwareServer_0079><br><P><br><MSL_ML sell_limit_order><br><SearchQuoteServer_0073> |

8/31/99

11

2W Systems Co., Inc.

## Selected OO Concepts and their UML notations

- Classes
- Objects and Composite Objects
- Associations, Constraints, and Comments
- Methods
- Events

8/31/99

12

2W Systems Co., Inc.

## Classes

- Semantics
- A *class* is a description of a set of objects that share the same attributes, operations, methods, relationships, and semantics. A class may use a set of interfaces to specify collections of operations it provides to its environment.

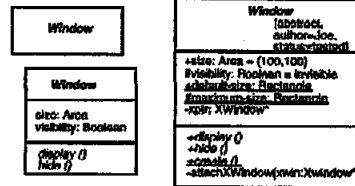
8/31/99

13

2W Systems Co., Inc.

## Classes

### ■ Notation

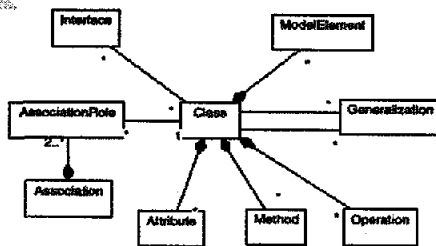


8/31/99

14

2W Systems Co., Inc.

## Classes



8/31/99

15

2W Systems Co., Inc.

## Classes

- The purpose of a *class* is to declare a collection of methods, operations, and attributes that fully describe the structure and behavior of objects. All objects instantiated from a class will have attribute values matching the attributes of the full class descriptor and support the operations found in the full class descriptor.

8/31/99

16

2W Systems Co., Inc.

## Objects

- Semantics
- An object represents a particular instance of a class.
- It has an identity and attribute values.

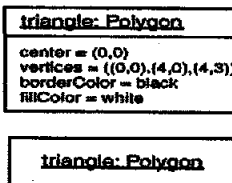
8/31/99

17

2W Systems Co., Inc.

## Objects

- UML Notation:  
*nhactname:classname*



8/31/99

18

2W Systems Co., Inc.

## Composite Object

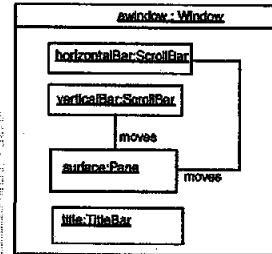
- A composite object represents a high-level object made of tightly-bound parts. This is an instance of a composite class, which implies the composition aggregation between the class and its parts..
- A SAS/AF frame is a typical composite object

8/31/99

19

2W Systems Co., Inc.

## Composite Object



8/31/99

20

2W Systems Co., Inc.

## Methods

- Semantics
- A *method* is the implementation of an operation. It specifies the algorithm or procedure that effects
- the results of an operation.

8/31/99

21

2W Systems Co., Inc.

## Events

- An *event* is the specification of a significant occurrence that has a location in time and space.
- An instance of an event can lead to the activation of a behavioral feature in an object.

8/31/99

22

2W Systems Co., Inc.

## Constraints, Associations & Comments

- Semantics
- A *constraint* is a semantic relationship among model elements that specifies conditions and propositions that must be maintained as true.
- A *comment* is a text string (including references to human-readable documents) attached directly to a model element.

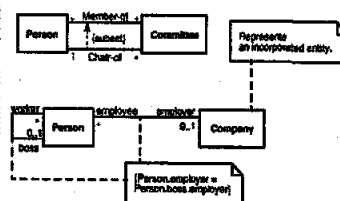
8/31/99

23

2W Systems Co., Inc.

## Constraints & Comments

- UML notation

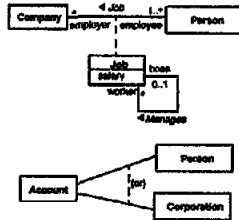


8/31/99

24

2W Systems Co., Inc.

## Associations



8/31/99

26

2W Systems Co., Inc.

## Diagrams

- Use-case diagrams: used to extract user requirements
- Sequence diagrams
- Context Diagrams: used to establish scope
- Event or Activity Diagrams: used to extract user requirements
- Class diagrams identify objects and their associations

8/31/99

26

2W Systems Co., Inc.

## Diagrams

- interaction diagrams document object interactions
- State-Transition or statechart diagrams document Object changes
- Implementation diagrams:
- component diagram
- deployment diagram

8/31/99

27

2W Systems Co., Inc.

## Actors

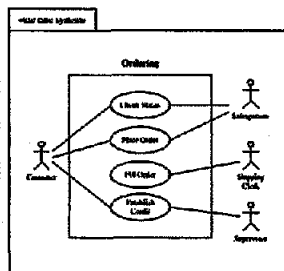
- An actor is a role that a user plays with respect to the system
- An actor is external to the system
- An actor does not have to be human: it can be an external system

8/31/99

28

2W Systems Co., Inc.

## Use-Case diagram



8/31/99

29

2W Systems Co., Inc.

## Sequence Diagram

- A sequence diagram represents an interaction, which is a set of messages exchanged among objects, against a vertical time axis

8/31/99

30

2W Systems Co., Inc.



## Deployment Diagram

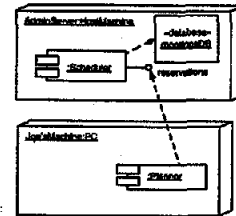
- Deployment diagrams show the configuration of run-time processing elements and the software components, processes, and objects that live on them.
- It will describe, for example, a three-tier architecture

8/31/99

37

2W Systems Co., Inc.

## Deployment Diagram



8/31/99

38

2W Systems Co., Inc.

## Benefits of UML

- OO is already the standard for most new shelf product development development
- Many companies are incorporating the UML as a standard into their development process and products, which cover disciplines such as business modeling, requirements management, analysis & design, programming, and testing.

8/31/99

39

2W Systems Co., Inc.

## Benefits of UML

- Understand the big picture (important for the SAS community who defines itself as Global Solution Providers)
- Excellent tool to communicate with all team members (Domain Experts, users, other developers)
- Cuts code, especially for large systems such as datawarehouses
- Quality Documentation

8/31/99

40

2W Systems Co., Inc.

## Drawbacks

- Challenge
- Learning curve
- Obstacles inherited from legacy culture
- Not a standard yet for application development in large MIS shops
- Corporate constraints on resources
- Not necessarily cuts time
- Not always compatible with RAD

8/31/99

41

2W Systems Co., Inc.

## Case Tools

- <http://www.qucis.queensu.ca/Software-Engineering/vendor.html>
- Rational ROSE from Rational Corp.
- WithClass and CRC Pro from MicroGold Software
- All do some degree of Reverse Engineering
- Come with various modules for code generation
- Maybe SAS will provide a CASE tool of its own, or in partnership with another vendor

8/31/99

42

2W Systems Co., Inc.

## OO Programming Languages

- C++
- Visual Basic, Forte
- Java
- PowerBuilder
- SmallTalk (IBM)
- Eiffel
- And of course SAS ...

8/31/99

43

ZW Systems Co., Inc.

## SAS→ as an OO Programming Tool

- Version 6.04 SAS/AF (PC/DOS) introduced SCL: the Screen Control Language, but it was not OO.
- Version 6.08: Introduction of frames and methods was a first step, but was still far from industry standards. It was mostly a GUI to build GUIs.

8/31/99

44

ZW Systems Co., Inc.

## SAS→ as an OO Programming Tool

- Version 6.10 through 6.12 (Orlando Release): many OO features:
- Class libraries and Viewers
- Data Table & Data form
- Process Flow Diagram
- Powerful lists

8/31/99

45

ZW Systems Co., Inc.

## SAS→ as an OO Programming Tool

- Version 7: (Nashville Release)
- SAS SCL is renamed SAS Component Object Model (SCOM)
- Components enable developers to perform complex tasks without code, and deliver applications that have the look and feel of the native host platform, and are virtually Plug and Play

8/31/99

46

ZW Systems Co., Inc.

## SAS→ V7 as an OO Programming Tool

- the complete Version 6 class library
- Expanded class library (Attribute linking, Drag and Drop, etc..)
- general-purpose classes, including collection classes, a halt handler class.
- development tools such as the Class Browser and SCL Static Analyzer
- non-visual models that provide access to SAS tables and external files

8/31/99

47

ZW Systems Co., Inc.

## The challenge for us

- A good software system is a software which was designed, developed and implemented within a rigorous methodology
- My purpose was to stimulate your curiosity and provide an eye opener for the emerging standard technology for Information Engineering, which we, SAS users, should not overlook.

8/31/99

48

ZW Systems Co., Inc.

## Bibliography

- [Booch et al.] Grady Booch, Jim Rumbaugh, and Ivar Jacobson, *Unified Modeling Language User Guide*, Addison Wesley, 1997.
- [Fowler] M. Fowler with K. Scott, *UML Distilled: Applying the Standard Object Modeling Language*, ISBN 0-201-32563-2, Addison-Wesley, 1997.
- *Object Magazine*
- *Journal of Object-oriented Programming*

8/31/99

48

2W Systems Co., Inc.

**Appendix**  
**Object-Oriented Programming**  
(Excerpt from SAS Doc-Online v7.1)

**Object-oriented programming (OOP)** is a technique for writing computer software.

The term **object oriented** refers to the methodology of developing software in which the emphasis is on the data, while the procedure or program flow is de-emphasized. That is, when designing an OOP program, you do not concentrate on the order of the steps that the program performs. Instead, you concentrate on the data in the program and on the operations that you perform on that data.

Advocates of object-oriented programming claim that applications that are developed using an object-oriented approach

- are easier to understand because the underlying code maps directly to real-world concepts that they seek to model
- are easier to modify and maintain because changes tend to involve individual objects and not the entire system
- promote software reuse because of modular design and low interdependence among modules
- offer improved quality because they are constructed from stable intermediate classes
- provide better scalability for creating large, complex systems.

**Object-oriented application design** determines which operations are performed on which data, and then groups the related data and operations into categories. When the design is implemented, these categories are called **classes**. A class defines the data and the operations that you can perform on the data. In SCL, the data for a class is defined through the class's **attributes, events, event handlers, and interfaces**. (Legacy classes store data in **instance variables**.) The operations that you perform on the data are called **methods**.

**Objects** are data elements in your application that perform some function for you. Objects can be **visual** objects that you place on the frame—for example, icons, push buttons, or radio boxes. Visual objects are called **controls**; they display information or accept user input.

Objects can also be **nonvisual** objects that manage the application behind the scenes; for example, an object that enables you to interact with SAS data sets may not have a visual representation but still provides you with the functionality to perform actions on a SAS data set such as accessing variables, adding data, or deleting data. An **object** or **component** is derived from, or is an **instance** of, a class. The terms object, component, and instance are interchangeable.

Software objects are self-contained entities that possess three basic characteristics:

- |          |                                                                                                                                                       |
|----------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| behavior | a collection of operations that an object can perform on itself or on other objects. <b>Methods</b> define the operations that an object can perform. |
| state    | a collection of attributes and their current values. You can set these values through SCL.                                                            |
| identity | a unique value that distinguishes one object from another. This identifier is referred to as its <b>object identifier</b> .                           |