OBJECT-ORIENTED ANALYSIS

Second Edition
by Peter Coad and Edward Yourdon

Yourdon Press, Prentice Hall Building,
Englewood Cliffs, New Jersey 07632 (1991) 233p.
ISBN 0-13-629981-4

Object orientation (OO) is present in several IT fields of activity in **universities and** industries, such as programming languages, expert systems building **tools**, systems analysis and, recently, databases design. **Object-oriented programming languages (OOPL)** originated during bringing forward such concepts as "encapsulation" **of Algol** 60, "object" of Simula 67. The Xerox-developed Smalltalk of mid70's is the **prototype of** a (symbolic) OOPL which uses objects, classes and inheritance properties. C++ of Bell Labs. or Objective C of **Productivity Tools** - recent extensions of **algorithmic** languages - are just good evidence that **OO** entered the world of **programming** languages. In the related field of Expert Systems generic building tools, KEE of Intellicorp., ART of Inference Corp. and Knowledge Kraft of Carnegie Group, all developed in the mid'80s, marked successful attempts at applying **OO** to expert systems, even though, sometimes (see S1 of Technowledge), the term of "object" was abusively used. **Entity Relationships** and **Semantic Data Modelling** methods apply OO concepts to systems analysis of (data-oriented) systems. Terms such as object, attributes and inheritance have been largely invoked in describing more (mathematical) model-oriented systems in CIM and manufacturing.

The intention of the authors of this book was to propose an object-oriented method of *system analysis* as the first step in an **object-oriented** integrated approach to **analysis, design, implementation** and **testing** (OOA, OOD, OOI and OOT). The OOD issues are deferred to be presented in another book [1]. The authors advocate for an **Object-Oriented Analysis** (OOA) not only as theory but also as practice, the practising system analysts being primarily addressed by the book. In the method proposed "objects as abstractions of the real world focus on significant aspects of the problem domain and the system's responsibilities", so that "a tangible, reviewable and manageable collection of model layers (Subject, Class & Object, Structure, Attribute and Service)" should be produced by the corresponding activities (page 5).

The book may be viewed as being composed of four parts such as introduction to OOA (Chapters 0-2), major activities of OOA (Chapters 3-7), related topics (Chapters 8-10) and appendices.

The **first part** sets the stage for OOA presentation

In "Introduction" the authors start with the methods of organisation the people employ in apprehending the real world (differentiation of experience into particular objects and their attributes, a distinction between whole objects and their component parts, and formation and distinction between different classes) and try to persuade the reader in considering OOA as a mature enough approach, supported by adequate **Object-Oriented Programming Languages** (OOPL), to the systems of today characterized by significant volatility, complexity and data orientation.

Chapter 1 entitled **"Improving Analysis"** defines the key concepts of "problem domain" and "system's responsibilities" and discusses other analysis challenge issues such as "communication", "continual change" and "reuse". Next, several principles for managing complexity such as procedural and data abstraction, encapsulation, inheritance association, communication with messages, scale, methods of organization in apprehending the real world, and behaviour categories are set forth. The major analysis approaches, such as *Functional Decomposition, Structured Analysis* (Data Flow Approach) and *Information Modelling* fall under critical survey with a view at introducing object oriented approach, as the only method that incorporates *all* the principles for managing complexity.

Chapter 2 "Experiencing an Object Perspective" introduces the main concepts of the object-oriented approach: objects, messages, classes, inheritance and illustrates them by examples in Smalltalk, an OOPL. This reviewer feels that a parallel exemplification with frame-based expert systems could help complete the terminology mappings.

The kernel of the book is contained in the second part which gradually introduces and covers the five major activities of OOA: finding **Classes & Objects (Chapter 3)**, identifying "General-Specialized" and "Whole-Part" **Structures (Chapter 4)**, discovering **Subjects (Chapter 5)**, defining **Attributes and Instance Connections (Chapter 6)** and defining **Services and Message Connections (Chapter 7)**. In a parallel presentation, each chapter has four sections "What" (definitions), "Why" (motivations), "How" (notation, identification, recommendations, special cases, etc.) and "Key points" summarizing the activity (notation and strategy). In order to illustrate particular concepts, ideas and recommendations of each activity, the authors use as a vehicle the same three examples: a) sensor system, b) registration and title system, and c) real-time airlift system.

Chapter 8 offers several selection criteria for **CASEs** to support OOA together with a list of existing products (including OOATool, an Object International product).

Chapter 9 goes on by introducing several preliminary issues concerning the **object oriented design** among which there are the new components of a multilayer, multicomponent model, that is "Human Interaction", "Task Management" and "Data Management" along with "Problem domain" component where the results of OOA appear.

Chapter 10 entitled **"Getting stated with OOD"** is about those key issues which are introducing OOA into an organization and mainly refer to the maturity of Object oriented techniques and to organization-needed sophistication.

Appendix A contains a summary of notations and strategies including an OOA model of OOA itself.

Appendix B presents the mapping of OOA to **OOD-STD-2167A**, a standard for planning and controlling software development in large software development projects.

This is definitely an excellent introduction to OOA.

Concepts and ideas are clearly brought about in presentation (sometimes colloquially). Particular attention is paid to using accurate terms, to making references to various Encyclopaedias and to other consacrated books as well.

By consistently presenting the same three application examples, by enriching their

annotation and contents from one chapter to another the reader is offered the red thread of the book. An apparent redundancy, due to the repetition of the key concepts at different times helps the reader keep in mind the material studied in the previous chapters.

Two final remarks are to be made.

1. The book is just one first step in the use of OOA. The software tool (OOATool) as well as other documentations such as the twin book [1] and "The Coad's Letters: New Advances in Object Analysis and Design" or Yourdon's "American Programmer" Journal, the authors are ready and willing to provide the reader with, for strengthening their orientation, are to be taken into consideration by the object oriented enthusiasts.

2. This reviewer feels that the approach made in the book is quite suitable especially for "data-oriented systems". As to (mathematical) "model-oriented systems" (such as **Decision Support Systems-DDS**), an adaptation effort of the techniques proposed in the book is to be expected.

REFERENCE

[1] COAD, P., and YOURDON, E., **Object oriented Design**, Prentice Hall, 1991.

F.G. Filip