Human IT

Tidskrift för studier av IT ur ett humanvetenskapligt perspektiv

Object Orientation as seen from library service

av Larserik Lönn

Abstract

This article is about the paradigm of object orientation as used in the context of systems analysis, systems design and computer programming. The paradigm is optimistically believed to be the silver bullet for the software crisis. But with the experience from library science, especially from the area of classification and indexing, we can reveal qualities in object orientation, which may overturn the optimistic believes to have found the silver bullet.

Innehåll

1. Object Orientation

Object oriented analysis, design and programming is said to provide the solution for many problems in systems development and programming. Rob Mattison writes in his book The Object Oriented Enterprise[1] that *"object orientation is the first methodology with an universal applicability", "a universal template that can be* *applied to all aspects of computer systems development"* and *"since the principles can be applied to all phases and aspects of a system, they can serve as the one true glue that can hold our systems together".*

Martin/Odell[2] states that OO technique will change *"the entire software industry, the way we use computers and networks, the way we analyze and design systems, the way we re-engineer corporations and the job of all I.S. professionals"* and *"object-oriented modeling and design is an integrating paradigm that should tie together all powerful tools and techniques for software creation".*

As stated above, the OO paradigm is a universal paradigm for all kind of systems and for all phases and aspects of systems development. A true revolution in the way we now often analyze reality and create systems.

Let us look at a few key concepts that builds up the OO technology, and compare them to the experience of application of these concepts from other areas. The first and most obvious concept, is the concept of an object. Martin/Odell(2) says that:

> "an object is any thing, real or abstract, about which we can store data and those methods that manipulate the data".

Booch[3] describes an object as a:

> " tangible entity that exhibits some well-defined behavior" and can be "a tangible and/or visible thing, something that can be apprehended intellectually or something toward which thought or action is directed".

From above we can conclude that an object is something that exists in the real world (real or abstract), something that has a distinct behavior and is sharply defined from its environment. In OO technology we create software objects which sole purpose is to simulate the activities of some real-world object. There is a very strong connection between the real world objects, and the objects of a computer system. The objects of the computer system models real word objects, therefor it should be possible to look at the experiences from history of handling of real world objects, and draw conclusions about what can happen in an object-oriented computer system.

The second concept which is central to object orientation is the concept of encapsulation. With encapsulation is meant that the behavior (methods) of the object and the objects data is tied closely together. They are encapsulated, and the objects "should be viewed as autonomous and self-contained workers"[1]. This statement is important when we introduce the third and last concept, the concept of inheritance. Inheritance deals with the problem how to organize and manage a large collection of objects.

The idea behind inheritance is that objects exists in (or at least can be put into) hierarchical relationships to each other, and that an object subordinated to an other object has some common behavior. The ‘child’ inherit behavior from its ‘parent’. When we create a software object this quality is very useful. The methods for a hierarchy can be defined only once in a high level in the hierarchy, and be inherited to all subordinate levels of objects. And by applying syntax and execution rules we can find the way to the right method. With inheritance we put dependencies between objects at the ‘managing’ level, which contradicts to some extent the concept of the autonomous object as encapsulation imply.

(Åter till början av artikeln)

2. Object classification

To organize and manage objects had its starting point in ancient times. Aristotle, the disciple of Plato, became famous for the creation of tools for the mind, an almost perfect logic for classes [4] and a methodology on how to create systems for classification. He also started to develop a botanical hierarchical system, but this work was never completed.

Carl von Linné (18.th century) developed his botanical system (also a hierarchical system) based on species. The hierarchies worked well until one of his students found a new flower, pleoria, which did not fit into the hierarchies. The flower seemed to belong to more than one hierarchy.

In the 19.th century librarians started to be interested in the classification of knowledge, with the main purpose to be able to find a system to retrieve information about books:

> "Classification is the act of grouping like things together, and the members share at least one characteristic which members of other classes do not possess. The ‘things classified may be concrete entities, ideas of such entities or abstractions" [5].

With our broad definition of the concept object, there is no doubt that a book can be an object, but what about knowledge represented in books. Knowledge can be considered as abstract representations of pieces of the real world, and the objects can be either the representation of the objects in the real world, or representations of the representations of objects in the real world. Consider as an example the area of AI, where the objects direct simulate knowledge, which not is the real world but mimic the real world. The conclusion from this will be that the area of classification of knowledge is relevant when we deal with the problem of organizing and managing objects.

To start with a classical classification system, take the Dewey Classification. It is a hierarchical system with a few main classes which are subordinated in smaller groups (or classes) to a number of levels. But since the principle of hierarchical systems are too simple to represent all types of relations that exist in the real world, the system is enhanced with notation principles where:

> "a given subject may appear in any number of disciplines" [6]. "The result of classification is the display of a network of relationships"[5].

An object in OO terminology will have more than one object (on the super level) to inherit from.

Professor Ranganathan [7], born 1892 in India, saw the problems with the hierarchical systems.. He says that the decimal classification (Dewey and other systems):

> "its attempts to enumerate all possible subjects and to provide a monolytic schedule to

> represent them caused great rigidity"[7].

Ranganathan developed the colon classification (CC) which viewed a subject capable of being analyzed into Basic Subjects and Isolates. The Isolates can be seen as different aspects on the subject. In his second version (1949) of CC he had generalized the Isolates to five: Personality, Matter, Energy, Space and Time. The result of this will result in a network of relationships, where each node is connected to five other nodes (the aspects). Again we can se the problem to identify a unique place for a subject (object) in an hierarchy. We often need to observe an object from different viewpoints (facets). In OO terminology we can se again that inheritance from more than on object is necessary, to be able to correct represent the complex real world.

The conclusion of what has been done in the area of classification is, that the problem of classification has been actual at least 2300 years. During this time no one has succeeded in the organizing of objects into strict hierarchies, other than for small groups of objects.

(Åter till början av artikeln)

3. Watch the steps

If object-oriented methodologies are going to succeed, we have to learn from the very long history and experience in classification theory and practice. To build object hierarchies (classification systems) which reflect a complex real world is very intricate. There are logical issues as well as time dependence and even cultural issues, and what we can expect to come out, as a result, is a network of relationships. The implication for OO will be that the vision of "universal applicability", "the glue that can hold our systems together" and " the integrating paradigm" will prove to have many obstacles in the way, maybe in the magnitude which classification theory and practice has proved. We shall be aware of that the powerful concept of inheritance, which implies organization of objects into hierarchies, in practice have strong limitations and demands the functionality of multiple inheritance to be able to handle even relatively simple object structures. But:

> "the inclusion of the ability to handle multiple inheritance makes the job infinitely more complex. It is tough enough to make single inheritance an efficient operation without adding the logical and physical consequences of multiple inheritance, with the astronomical increase in system overhead that this capability would entail"[1].

(Åter till början av artikeln)

Om författaren

Larserik Lönn är universitetslektor i informatik vid Institutionen för Data och Affärsvetenskap (IDA) vid Högskolan i Borås och doktorand i ämnet 'Informatik med Systemvetenskap' vid Stockholms universitet. LL har tidigare varit aktiv inom näringslivet och som lektor vid Bibliotekshögskolan, med inriktning mot biblioteksautomation och system för informationsåtervinning. Hans nuvarande forskning är inriktad mot systemutveckling och simulering av organisationer som generella system

Fotnoter/Referenser

1. **Mattisen** R, Sipolt M.J.: The Object-Oriented enterprise - Making Corporate Information Systems Work, McGraw-Hill 1994, ISBN 0-07-041031-3. Åter till texten

2. **Martin** J, Odell J. J.: Object Oriented Analysis & Design, PTR Prentice Hall 1992, ISBN 0-13-630245-9. Åter till texten

3. **Booch** Grady: Object Oriented Design and Applications, Benjamin-Cummins Publishing Co. 1991. Åter till texten

4. **Lukasiewicz** J.: Aristotle's Syllogistics from the standpoint of Modern Formal Logic (1975). Åter till texten

5. **Buchanan**, B.: Theory of Library Classification, Bingley London (1979). Åter till texten

6. **Blomberg** M, Weber H.: An Introduction to Classification and Number Building in Dewey. Libraries Unlimited Inc. (1976). Åter till texten

7. **Ranganathan** S. R.: Impact of growth in the universe of subjects on classification in International Federation for Documentation, Committee on Classification Research, FID/CR Report No 12 (1972), FID Publ. Series no 405. Åter till texten

---

Åter till Human IT 3/1998