About OOP and Other Programming Paradigms

Adrian Kosmaczewski

2006-05-10

Does OOP reflect a "natural" way of thinking? Is it a better choice than the procedural programming paradigm?

In computer science, to say that one approach is "better" than another is to miss a great detail: I do not think that there are "better" or "natural" paradigms per se, but just apppropriate answers to certain problems in a given context.

In his 1962 book "The Structure of Nature Revolutions", Thomas Kuhn introduces the idea of the "paradigm shift"; following this idea, human knowledge does not evolve gradually, but rather in discontinuous jumps, called "paradigm shifts" or "scientific revolutions":

> "A scientific revolution occurs, according to Kuhn, when scientists encounter anomalies which cannot be explained by the universally accepted paradigm within which scientific progress has thereto been made. The paradigm, in Kuhn's view, is not simply the current theory, but the entire worldview in which it exists, and all of the implications which come with it. There are anomalies for all paradigms, Kuhn maintained, that are brushed away as acceptable levels of error, or simply ignored and not dealt with (a principal argument Kuhn uses to reject Karl Popper's model of falsifiability as the key force involved in scientific change)."

(Wikipedia, 2006)

In the case of the activity of software engineering, the paradigm shift from procedural to object-orientation is quite evident, both historically and technically speaking.

The first programming language to introduce the concept of object-oriented programming was Simula, at the beginning of the 60s; Simula (http://www.engin.umd.umich.edu/CIS/course.des/cis400/simula/simula.html, http://staff.um.edu.mt/jskl1/talk.html), as the name implies, was developed by scientist of the Norwegian Computing Center in Oslo, to simulate real-life events; thus, Simula was designed with a special objective and problem context in mind.

10 years after Simula, the Smalltalk (http://www.smalltalk.org/) programming language was created by Alan Kay in the Xerox Palo Alto Research Center (Xerox PARC, http://www.parc.com/) in Silicon Valley, to support the first computer featuring a full GUI (Graphical User Interface), the "Alto". The Smalltalk language and the Alto computer were both groundbreaking at the time (around 1973) and they paved the way to the modern personal computer as we know it today. Moreover, the object-oriented paradigm appeared as the most appropriate one to design a "windowed" system, with buttons and menus, operated by a mouse. This fact, coupled with the popularity of the C++ programming language since the 80s, set the tone for the mainstream trend in the creation of software until the appearance of Java in 1995.

Having said this, it is worth noting that the first programming toolkit for the Macintosh was designed to be used with the Pascal programming language (one of the most well-known procedural programming languages):

> "Parts of the original Macintosh operating system were written in Pascal and Motorola 68000 assembly language (though later versions incorporated substantial amounts of C++ as well), and the most frequent high-level language used for development in the early Mac community was Pascal."

(Wikipedia, 2006)

It is, then, difficult to talk about a "better" approach, when comparing OOP to procedural programming; actually, each problem must be evaluated in detail to find the paradigm that gives the best answer, not only in terms of performance and compatibility, but also in terms of maintainability.

Finally, it must be said that procedural and object-orientation are not the only programming paradigms:

- Structured programming - compared to Unstructured programming
- Imperative programming, compared to Declarative programming
- Message passing programming, compared to Imperative programming
- Procedural programming, compared to Functional programming
- Value-level programming, compared to Function-level programming
- Flow-driven programming, compared to Event-driven programming
- Scalar programming, compared to Array programming
- Class-based programming, compared to Prototype-based programming (within the context of Object-oriented programming)
- Constraint programming, compared to Logic programming
- Component-oriented programming (as in OLE)
- Aspect-oriented programming (as in AspectJ)
- Rule-based programming (as in Mathematica)
- Table-Oriented Programming (as in Microsoft FoxPro)
- Pipeline Programming (as in the UNIX command line)
- Post-object programming
- Subject-oriented programming

- Reflective programming
- Dataflow programming (as in Spreadsheets)
- Policy-based programming
- Annotative programming - http://www.flare.org

(Wikipedia, 2006)

References

Smalltalk History [Internet], http://www.smalltalk.org/smalltalk/history.html
(Accessed February 26th, 2006)

Wikipedia, “Pascal programming language” [Internet], http://en.wikipedia.org/wiki/Pascal_programming_lan
(Accessed February 26th, 2006)

Wikipedia, “Programming paradigm” [Internet], http://en.wikipedia.org/wiki/Programming_paradigm
(Accessed February 26th, 2006)

Wikipedia, “Thomas Kuhn” [Internet], http://en.wikipedia.org/wiki/Thomas_Kuhn
(Accessed February 26th, 2006)