Super-Object-Oriented Programming and Model Nesting

Eugene Kindler, University of Ostrava, Czech Republic

The year 2007 is a jubilee year – 40 years ago, in 1967, the Norwegian programming language called SIMULA 67 was presented to the representatives of the world professional community (at the IFIP working Conference on Simulation Programming Languages). The language offered all tools covered later by the term Object-Oriented Programming, i.e.

(1) classes as encapsulation of data and procedures,
(2) subclasses as enrichment of classes by new data and new procedures, and
(3) virtuality as a possibility to get procedures new contents in the subclasses.

But SIMULA offered other tools that were not covered by the term object-oriented languages and that were ignored by many famous object-oriented programming languages as C++, Eiffel, (new versions of) Pascal and SmallTalk. Later, these tools were characterized as Super-Object-Oriented Programming. They are

(4) including “life rules” into the classes,
(5) assigning virtuality for the targets of program transfers,
(6) including scheduling statements into the life rules,
(7) generalizing the scheduling to quasiparallel control of life rules performance,
(8) including complete block structuring,
(9) enabling classes as local in blocks,
(10) enabling classes to carry declarations of other classes.

(4) together with (6) enabled SIMULA to be used as a process-oriented simulation language. (9) enabled to simulate (or – in general – to model) systems containing „intelligent“ elements, i.e. containing elements, in the life of which phases arise and sink, dzuring which they apply their proper „notions“ (represented by the classes local in the blocks nested in the life rules). (10) is similar to (9) and enables consider class instances as cerriers of formal theories (world viewings, formal languages), (7) makes SIMULA a general purpose language, passing the domain of simulation over.

The main application of the mentioned tools is nesting of simulation models, i.e. simulation models of systems that contain elements able to simulate (namely computers or imagining persons). Such applications will be presented at the tutorial, namely those concerning transport systems (dynamics in container terminals, public personal transport, operational transport in production halls), some mentions will be made on patients dynamics in hospitals, on rectification column simulation, on demographic prediction of regions and on special very effective optimizing software enabling to go to the optimum during parallel simulation of several dynamically modifying variants.

Only some tools of the list (4)-(10) can be observed at a small number of quite new programming languages like BETA, JAVA and MODSIM III.

The core of the tutorial will be oriented to the language SIMULA itself, to its exact rules for programming, to recommendation how to use them in nesting models and to certain relations to other programming languages. The participants will get a free SIMULA compiler for PC under Windows).

Some References:

E. Kindler: SIMULA and Super-Object-Oriented Programming. In: O.Owe, S. Krogdal, T. Lychne (eds.): *From Object-Orientation to Formal Methods* [Lecture Notes in Computer Science, vol. 2635]. Springer, Berlin, Heidelberg, New York, 2004, pp 165-182. ISBN 3-540-21366-X, ISSN 0302-9743

E. Kindler, T. Coudert, P. Berruet: Component-Based Simulation for a Reconfiguration Study of Transitic Systems, *SIMULATION*, Vol. 80, 2004, No. 3, pp.153-163. ISSN 0037-5497

E. Kindler: Object-Oriented Representations of Formal Theories as Tools for Simulation of Anticipatory Systems. In: D. M. Dubois (ed.): *Computing Anticipatory Systems CASYS 2005 – Seventh International Conference, Ličge, Belgium, 8-13 August 2005*. American Institute of Physics, Melville, New York, 2006 [AIP Conference Proceedings Volume 839], pp. 253-259. ISBN 0-7354-0331-7, ISSN 0094-243X, L.C. Catalog Card No. 2006926134