INFORMATION PROCESSES, PROGRAMMING LANGUAGES AND PERSPECTIVES ON REALITY

K NYGAARD

Rapporteurs: Jason N Bain and Rogerio de Lemos

Information Processes, Programming Languages and Perspectives on Reality

K Nygaard

Norwegian Computer Society
and University of Oslo
Forskruisun 1B
Postboks 335 Blinden
Oslo 3
Norway

The lecture will approach the development of programming languages from the point of view that they represent a perspective on a class of phenomena in the real world. They provide a way of understanding and describing these phenomena. A description of some existing or conceived referent phenomenon may at the same time serve as a prescription (a program) for creating a structurally similar model phenomenon in a computer. Observation of the model provides information about the referent.

The model is a dynamic phenomenon: an information process, or in standard terms; a program execution. Its main aspects are: its substance (objects, files, records), its state (the state of the properties of the substance) and its transitions between states.

The perspective is characterised by a way of selecting those properties of the phenomena that are being considered (and, by implication, those that are ignored), and by providing concepts and other cognitions that are being used in the interpretation and description of the selected properties.

Initially the perspectives were strongly influenced by the way the computers were organised and thus how the model was understood. The main task was to describe the phenomenon in terms of the model. Later the emphasis has been on creating ways of describing the referent phenomena in terms of suitable perspectives for people.

The lecture will discuss some of these perspectives, both for specific languages and categories of languages. Important categories are: Transition oriented languages (including functional languages), state oriented languages (including constraint oriented languages) and substance oriented languages (object oriented languages).

DISCUSSION

Rapporteur: Ron Kerr

Lecture One

Professor Tanenbaum objected that Professor Nygaard's comparison of informatics with natural science was misleading. When God created the universe, like most implementors since then he did not bother to write any documentation about how it finally worked and people like physicists and chemists are performing reverse engineering to find out what the rules are. That is not true of computing science in which we actually design things ourselves. The correct analogy is with engineering. Building a software system is like building a bridge, nuclear reactor, etc. These are man-made objects and their construction is not related to fields like physics and chemistry in the natural sciences and the paradigms used there are inappropriate for computing science.

Professor Nygaard partly agreed and partly disagreed. He observed that a field of knowledge may have an aspect of construction. For example, chemical engineering is a major facet of chemistry which is founded on knowledge acquired by analysis and empirical study. He regards engineering as one aspect of a certain type of knowledge but accepted that informatics is strongly concerned with construction.

Professor Rabin observed that Professor Nygaard had introduced a great mass of concepts and relationships between them, all at an extremely sophisticated level. Comparing this with, for example, the fields of quantum mechanics and cognition, Professor Rabin suggested that these, among the most sophisticated areas of science, were founded upon fewer concepts, categories and intricate relationships than Professor Nygaard had presented and he questioned whether computing science was really so much more profound.

Professor Nygaard considered that Professor Rabin was over-stretching his analogy. He suggested that perspectives could be adopted in theoretical science such that certain issues, for example substance, were abstracted away. Since his starting point is phenomena, substance, state and transition are the fundamental concepts and these are not numerous. However, he would be happy to have superfluous concepts and issues brought to his attention.

Professor Nygaard added that an additional factor is that systems are employed in social contexts. This gives rise to the notion of conflict, something with which scientists are unhappy. Consequently, a proper approach to system development cannot avoid certain elements of the social sciences influencing some aspects of one's reasoning.

Lecture Two

Professor Turski invited comments on PL/1. Professor Nygaard repeated his belief that a programming language must be founded upon an idea or philosophy. That behind PL/1 was commercial exploitation. This was in contrast to, for example, C++ whose design was motivated by the need to have SIMULA's philosophy of software design and structure available within the C programming arena. Professor Nygaard recounted the tale of a young man who found himself a new girl friend when he moved to a new town. In reply to the question of what the new girl had that the old one did not, he replied, "Nothing, but she has it here!" C++ is rather a paradox. It is popular in certain quarters because it has C and unpopular in others for the same reason.