Rendez-Vous with Metric Semantics*

J. W. de BAKKER
CWI,
Postbus 4079, NL-1009 AB Amsterdam, Netherlands.
& Vrije Universiteit.
E. P. de VINK
Department of Mathematics and Computer Science,
Vrije Universiteit,
De Boelelaan 1081a, NL-1081 HV Amsterdam, Netherlands.

Received 21 October 1991
Revised Manuscript received 1 October 1992

Abstract A comparative semantic study is made of an element of the family of concurrent object-oriented programming languages. Particular attention is paid to two notions: (i) dynamically evolving process structures, including a mechanism to name and refer to processes and a means to create new processes, and (ii) rendez-vous between processes involving the sending and answering of messages and the induced execution of method calls. The methodology of metric semantics is applied in the design of operational and denotational semantics, as well as in the proof of their equivalence. Both semantics employ domains which are determined as fixed points of a contracting functor in the category of complete metric spaces. Moreover, fruitful use is made of the technique of defining semantic meaning functions as fixed points of contracting higher-order mappings. Finally, syntactic and semantic continuations play a pervasive role.

Keywords: Concurrency, Object-Oriented, Metric Semantics, Rendez-Vous, Process Creation, Continuations, Operational Semantics, Denotational Semantics.

§1 Introduction

We shall present a comparative semantic study of a language of the COOP (concurrent object-oriented programming) variety. Particular attention will be paid to the following two phenomana

* The research of J. W. de Bakker was partially supported by ESPRIT Basic Research Action 3020: Integration.
dynamically evolving process structures, including a mechanism to name and refer to processes and a means to create new processes;

- a version of rendez-vous between processes involving the sending and answering of messages and the ensuing execution of method calls.

The language we consider is a slightly simplified version of the language POOL—the parallel object-oriented language designed by America. Several semantic investigations of this language have appeared already: operational semantics, denotational semantics, and a comparison of these two. Cf. also Ref. 5 for a somewhat streamlined version of parts of Refs. 2, 3 and 27—excluding the more difficult sections of the comparison—and Ref. 7, where an improvement of POOL's denotational semantics which is organized in three layers (for statements, objects and programs) is described. The latter paper is intended as well as a contribution to the issue of the full abstractness of the POOL semantics.

The treatments in Refs. 3 and 27 are rather complex and demand much from the uninitiated reader. The first aim of the present paper is to provide a more comprehensible version of these investigations, with special emphasis on the comparative issues. Partly, this is achieved by a presentation in two stages, both dealing with dynamically evolving processes, but only in the second one with a facility to name and refer to processes. Also, a careful tuning of the design of the operational and denotational definitions—in particular by the systematic use of so-called syntactic and semantic continuations—results in a transparent view of the relationship between the two models. Maybe more importantly, we propose a substantial simplification in the way the rendez-vous concept is handled. Firstly, the operational semantics rule for the rendez-vous is now appealingly simple and, secondly, some of the complexities in the denotational models of Refs. 3 and 27, in particular in the definition of the merge operator, are to a large extent avoided. Related to this we find that the equation determining the domain used in POOL's denotational semantics is essentially simplified in our approach. (In the domain equation \( P = F(P) \), \( F(P) \) has no more subterms of the form \( (P \rightarrow \ldots) \). See Section 2 for background on this.) In addition, the somewhat extraneous use of the denotational meaning function \( D \) as part of the intermediate operational semantics in Ref. 27 is no more necessary.

The second aim of our paper is to provide a case study is the use of metric semantics. Let us first devote a few words to its basic principles. Consider two computations \( p_1, p_2 \). A natural distance \( d(p_1, p_2) \) may be defined in terms of the notion of initial segment \( p(k) \) of \( p \)—roughly, that part of \( p \) consisting of the first \( k \) steps (if present, otherwise \( p \) itself). Now we put \( d(p_1, p_2) = 2^{-n} \), where \( n \) is the length of the longest common initial segment of \( p_1 \) and \( p_2 \) (i.e., \( n = \sup\{k \mid p_1(k) = p_2(k)\} \)). Details vary with the form of the \( p_1, p_2 \). If computations are given as words (finite of infinite sequences of atomic actions), we take the standard notion of prefix; if \( p_1, p_2 \) are trees, we use truncation at depth \( k \) for