

Roles of Variables in Experts' Programming Knowledge

Jorma Sajaniemi¹ and Raquel Navarro Prieto²

¹ University of Joensuu, Department of Computer Science,
P.O.Box 111, 80101 Joensuu, Finland,
jorma.sajaniemi@joensuu.fi,

WWW home page: <http://www.cs.joensuu.fi/~saja/>

² Universitat Pompeu Fabra, Estacio de la Comunicacio,
Ocata 1, 08003 Barcelona, Spain

Abstract. Roles of variables capture the dynamic nature of variables, i.e., their behavior. Only ten roles are needed to cover 99 % of variables in novice-level procedural programs. Roles were originally identified by studying variables in existing programs and creating a classification for them. In order to find out whether roles are a part of experts' programming knowledge, we conducted a knowledge elicitation investigation where professional programmers studied programs and the resulting mental representations were elicited using card sorting and interviews. This paper presents the analysis of the results from the point of view of the role theory. All roles appearing in the materials were identified by participants. There was some variation in perceiving the nature of behavior from the lifetime of a variable and in considering the similarity of behaviors. The roles could however be easily found in the participants' card sorting results and in the dendrogram obtained by hierarchical cluster analysis.

1 Introduction

Programming involves the use of abstract concepts at various levels of abstraction. One such concept is the notion of variable plans, which represent stereotypic uses of variables [1, 2]. Based on the plan concept, Sajaniemi [3] has developed a theory of the roles of variables, which characterizes variables as having one of a set of distinctive roles. The role theory applies to small and large programs, and to different programming paradigms like functional and object-oriented programming. A set of ten roles covers practically all variables in novice-level procedural programs.

Sajaniemi and Kuittinen [4] have used roles in teaching elementary programming. Students were divided into three groups that were instructed differently: in the traditional way with no treatment of roles; using roles throughout the course; and using a role-based program animator in addition to using roles in teaching. The results indicate that students are not only able to understand the role concept and to apply it in new situations but that the roles provide students a new conceptual framework that improves program comprehension; and that the use of the animator elaborates this knowledge resulting in a richer set of programming plans and in skilled programming [4, 5].

In another study, Ben-Ari and Sajaniemi [6] tested the understandability and acceptability of the role concept and of the individual roles as seen by computer science

