Programming Languages and Systems for Prototyping Concurrent Applications

WILHELM HASSELBRING

Tilburg University

Concurrent programming is conceptually harder to undertake and to understand than sequential programming, because a programmer has to manage the coexistence and coordination of multiple concurrent activities. To alleviate this task several high-level approaches to concurrent programming have been developed. For some high-level programming approaches, *prototyping* for facilitating early evaluation of new ideas is a central goal.

Prototyping is used to explore the essential features of a proposed system through practical experimentation before its actual implementation to make the correct design choices early in the process of software development. Approaches to prototyping *concurrent* applications with very high-level programming systems intend to alleviate the development in different ways. Early experimentation with alternate design choices or problem decompositions for concurrent applications is suggested to make concurrent programming easier.

This paper presents a survey of programming languages and systems for prototyping concurrent applications to review the state of the art in this area. The surveyed approaches are classified with respect to the prototyping process.

Categories and Subject Descriptors: D.1.3 [**Programming Techniques**]: Concurrent Programming—*Parallel programming*; *Distributed programming*; D.2.1 [**Software Engineering**]: Requirements/Specifications; D.2.2 [**Software Engineering**]: Design Tools and Techniques—*Computer-aided software engineering* (CASE); *Petri nets*; *Software libraries*; D.2.6 [**Software Engineering**]: Programming Environments—*Interactive environments*; D.3.2 [**Programming Languages**]: Language Classifications—*Concurrent, distributed, and parallel languages*; *Very high-level languages*; D.3.3 [**Programming Languages**]: Language Constructs and Features—*Concurrent programming structures*

General Terms: Languages

Additional Key Words and Phrases: Concurrency, distribution, parallelism, rapid prototyping, very high-level languages

1. INTRODUCTION

During the last decades, particular attention has been focused on concurrent programming within the computer science community. A *concurrent* program specifies two or more processes that cooperate in performing a task [Andrews 1991]. Each process is a sequential pro-

Author's address: INFOLAB, Tilburg University, PO Box 90153, Tilburg, 5000 LE, The Netherlands; email: hasselbring@acm.org.