Chapter 12
Design and Implementation of an Autonomic Code Generator Based on RTPA

Yingxu Wang
University of Calgary, Canada

Xinming Tan
Wuhan University of Technology, China

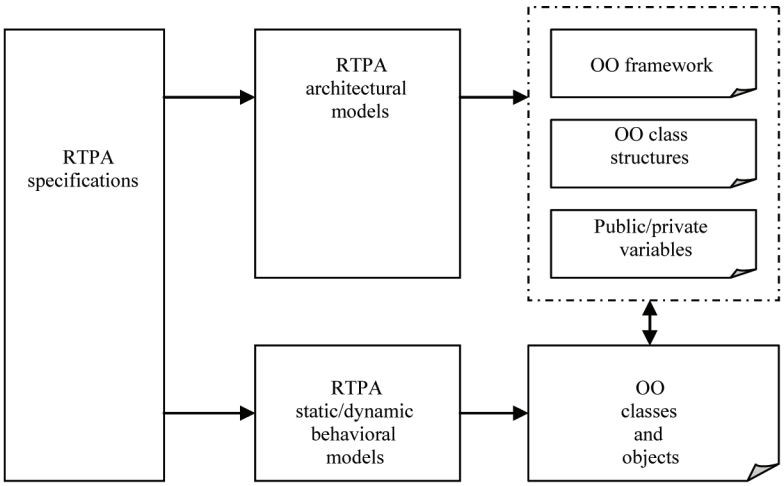
Cyprian F. Ngolah
Sentinel Trending & Diagnostics Ltd., Calgary, Canada

ABSTRACT

Real-Time Process Algebra (RTPA) is a denotational mathematics for the algebraic modeling and manipulations of software system architectures and behaviors by the Unified Data Models (UDMs) and Unified Process Models (UPMs). On the basis of the RTPA specification and refinement methodologies, automatic software code generation is enabled toward improving software development productivity. This paper examines designing and developing the RTPA-based software code generator (RTPA-CG) that transfers system models in RTPA architectures and behaviors into C++ or Java. A two-phrase strategy has been employed in the design of the code generator. The first phrase analyzes the lexical, syntactical, and type specifications of a software system modeled in RTPA, which results in a set of abstract syntax trees (ASTs). The second phrase translates the ASTs into C++ or Java based on predesigned mapping strategies and code generation rules. The toolkit of RTPA code generator encompasses an RTPA lexer, parser, type-checker, and a code builder. Experimental results show that system models in RTPA can be rigorously processed and corresponding C++/Java code can be automatically generated using the toolkit. The code generated is executable and effective under the support of an RTPA run-time library.

DOI: 10.4018/978-1-4666-0264-9.ch012

Figure 1. Program code generation based on RTPA



INTRODUCTION

Automatic code generation on the basis of rigorous system models and formal specifications is one of the central objectives of software engineering, which leads to the improvement of software development productivity, efficiency, and quality (McDermid, 1991; Michael & Butler, 1996; Wang, 2007). However, automatic code generation is an intricate and difficult endeavor in software engineering. Many efforts were reported on simplified system specifications and simple target languages (RAISE, 1995; Univan & Chris, 2000). It is a great challenge to design and implement a comprehensive code generator that covers the entire range of software modeling needs including real-time device drives and machine-level event/time/interrupt-driven mechanisms.

The process metaphor for software system modeling, specification, and implementation was initially proposed by Hoare and others (Hoare, 1978; Milner, 1980) that perceived a software system as the composition of a set of interacting processes.

Various algebraic approaches to describe the behaviors of communicating and concurrent systems were developed known as process algebra (Hoare, 1978, 1985; Milner, 1980), which provide a set of formal notations and rules for describing algebraic relations of software behaviors. A set of denotational mathematics was recently developed (Wang, 2008a) for rigorously modeling both architectures and behaviors of software systems. Denotational mathematics is a category of expressive mathematical structures that deal with *high-level mathematical entities* beyond numbers and sets, such as abstract objects, complex relations, perceptual information, abstract concepts, knowledge, intelligent behaviors, behavioral processes, and systems (Wang, 2009a). The paradigms of de-notational mathematics are such as concept algebra (Wang, 2008b), Real-Time Process Algebra (RTPA) (Wang, 2008c), and system algebra (Wang et al., 2008).

RTPA is a form of de-notational mathematics for formally modeling and describing architectures and behaviors of software systems (Wang, 2002, 2008c, 2008d). Based on the RTPA methodology and models, software code may be seamlessly generated as shown in Fig. 1. In the scheme of RTPA-based code generation, the RTPA architectural model for a system is used to generate the structural framework and global/local variables of classes or objects; while the RTPA behavioral