Cognitive Technologies

Petra Hofstedt

Multiparadigm Constraint Programming Languages

Springer

Prof. Dr. Petra Hofstedt
Brandenburgische Technische
Universität (btu) Cottbus
Fakultät 1
Lehrstuhl Programmiersprachen
und Compilerbau
Postfach 101344
03013 Cottbus
Germany
petra.hofstedt@informatik.tu-cottbus.de

Cognitive Technologies ISSN 1611-2482
ISBN 978-3-642-17329-5 e-ISBN 978-3-642-17330-1
DOI 10.1007/978-3-642-17330-1
Springer Heidelberg Dordrecht London New York

ACM Codes D.1, D.3, I.2

Library of Congress Control Number: 2011930425

Cover design: KunkelLopka GmbH, Heidelberg

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

For Sebastian and Pierre

Foreword

"Modeling" has become one of the primary concerns in modern Software Engineering. The reason is simple: starting development processes from clear and succinct models has proven to foster not only quality but also productivity. With the advance of modeling there also came a desire for automatic code generation from models. This way, the costly and error-prone implementation in terms of low-level languages should be saved. To this end, the models need to be "executable" in the widest sense.

In this general picture the concepts of constraint programming obtain a new and economically important role. Even though they are not in the current mainstream of UML-style graphical languages, they are extremely well suited for describing models. This is evident by considering the very nature of constraints: one formulates the properties that a software system shall fulfill; the implementation is done automatically by the constraint solver. So the time is ripe for constraint-based programming to come out of the more academic world, to which it still is constrained to a large extent, and show its potential for modeling real-world applications.

However, there is no silver bullet. Classical constraint systems in their pure forms are not expressive enough to be used in a large variety of application domains. Therefore they need to be augmented by other styles and concepts for programming and modeling. This leads into the realm of so-called "multi-paradigm languages". There have been all kinds of approaches in Computer Science to address the "no-silver-bullet" issue. Examples range from voluminous languages such as PL/1 or Ada (which failed miserably), huge libraries (which are non-standardized and thus lead to severe problems in the long run), or Microsoft's .Net approach (which solves the problem at least on the low level of machine code). The keyword DSLs (domain-specific languages) can be viewed as the general circumscription for all kinds of attempts to address the multiparadigm idea. By contrast to .Net the emphasis here is on the integration at the level at which the users formulate their intentions.

In this dynamic evolution of ideas, concepts and efforts, the book by Petra Hofstedt provides a valuable snapshot and assessment of the state of the art and the future directions for potential evolutions. The first part of the book gives an overview of paradigms, languages and compilers that are currently available both for academic experiments and for practical applications. Based on a sound description of the mathematical foundations, the general concepts of multiparadigm constraint languages are explained and the technical means for their implementation in the form of efficient solvers are presented. Of particular interest is the classification of the interplay of the different styles of programming, such as constraint logic programming, constraint imperative programming, constraint object programming, et cetera.

The second part of the book complements this overview of languages and concepts by looking at the application domains. However, the word "case studies" does not refer here to a collection of selected examples. Rather it refers to the techniques that are needed to realize some of the multi-paradigmatic compositions. This is done on two orthogonal dimensions: First, one of the most complex combinations of paradigms is presented quite concretely, namely concurrent constraint-functional programming. Second, a generic framework is elaborated, which shall make it relatively easy to compose all kinds of paradigms within new language shells. This is best expressed by the catch-phrase "from solver cooperation to language integration".

The book by Petra Hofstedt comes at a time of highly dynamic evolutions in Software and System Engineering, in particular with respect to modeling, specification and high-level problem description. It provides a valuable insight into one important aspect of this field, namely the activities centered around the declarative description of systems by way of their properties, that is, by way of constraints. It will help the reader to understand the foundations of the approach and to get a better judgement of the future perspectives.

Berlin, December 2010 Peter Pepper

Contents