1. Record Nr. UNINA9910483607703321

Titolo	Multiparadigm Programming in Mozart/Oz : Second International Conference, MOZ 2004, Charleroi, Belgium, October 7-8, 2004, Revised Selected Papers / / edited by Peter Van Roy
Pubbl/distr/stampa	Berlin, Heidelberg : , : Springer Berlin Heidelberg : , : Imprint : Springer, , 2005
Edizione	[1st ed. 2005.]
Descrizione fisica	1 online resource (XVI, 336 p.)
Collana	Programming and Software Engineering, , 2945-9168 ; ; 3389
Altri autori (Persone)	Van-RoyPeter
Disciplina	005.13
Soggetti	Compilers (Computer programs) Computer science Software engineering Computer programming Operating systems (Computers) Compilers and Interpreters Computer Science Logic and Foundations of Programming Software Engineering Programming Techniques Operating Systems
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Note generali	Bibliographic Level Mode of Issuance: Monograph
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Keynote Talk -- The Development of Oz and Mozart -- Security -- The Structure of Authority: Why Security Is Not a Separable Concern -- The Oz-E Project: Design Guidelines for a Secure Multiparadigm Programming Language -- Computer Science Education -- A Program Verification System Based on Oz -- Higher Order Programming for Unordered Minds -- Software Engineering -- Compiling Formal Specifications to Oz Programs -- Deriving Acceptance Tests from Goal Requirements -- Human-Computer Interfaces and the Web -- Using Mozart for Visualizing Agent-Based Simulations -- Web Technologies for Mozart Applications -- Overcoming the Multiplicity of Languages and Technologies for Web-Based Development Using a Multi-paradigm Approach -- Distributed Programming -- P2PS: Peer-to-Peer

Development Platform for Mozart -- Thread-Based Mobility in Oz -- A Fault Tolerant Abstraction for Transparent Distributed Programming -- Grammars and Natural Language -- The CURRENT Platform: Building Conversational Agents in Oz -- The Metagrammar Compiler: An NLP Application with a Multi-paradigm Architecture -- The XDG Grammar Development Kit -- Constraint Research -- Solving CSP Including a Universal Quantification -- Compositional Abstractions for Search Factories -- Implementing Semiring-Based Constraints Using Mozart -- A Mozart Implementation of CP(BioNet) -- Constraint Applications -- Playing the Minesweeper with Constraints -- Using Constraint Programming for Reconfiguration of Electrical Power Distribution Networks -- Strasheela: Design and Usage of a Music Composition Environment Based on the Oz Programming Model -- Solving the Aircraft Sequencing Problem Using Concurrent Constraint Programming -- The Problem of Assigning Evaluators to the Articles Submitted in an Academic Event: A Practical Solution Incorporating Constraint Programming and Heuristics -- An Interactive Tool for the Controlled Execution of an Automated Timetabling Constraint Engine.

Sommario/riassunto

To many readers, Mozart/Oz represents a new addition to the pantheon of p- gramming systems. One way of evaluating a newcomer is through the eyes of the classics, for example Kernighan and Pike's "The Practice of Programming," a bookthatconcludeswithsix"lastingconcepts": simplicityandclarity,generality, evolution,interfaces,automation, andnotation.KernighanandPikeconcentrate on using standard languages such as C and Java to implement these concepts, but it is instructive to see how a multiparadigm language such as Oz changes the outlook. Oz's concurrency model yields simplicity and clarity (because Oz makes it easier to express complex programs with many interacting components), g- erality, and better interfaces (because the data'ow model automatically makes interfaces more lightweight). Constraint programming in Oz again yields simplicity and clarity (because theprogrammercanexpresswhatneedstobetrueratherthanthemorecomplex issue of how to make it true), and o'ers a powerful mathematical notation that is di'cult to implement on top of languages that do not support it natively. Mozart's distributed computing model makes for improved interfaces and eases the evolution of systems. In my own work, one of the most important concernsistobeabletoquicklyscaleupaprototypeimplementationintoalar-scale service that can run reliably on thousands of computers, serving millions of users. The ?eld of computer science needs more research to discover the best ways of facilitating this, but Mozart provides one powerful approach. Altogether, Mozart/Ozhelpswithallthelastingconceptsexceptautomation, and it plays a particularly strong role in notation, which Kernighan and Pike pointoutisanunderappreciatedarea. Ibelievethatprovidingtherightnotation isthemostimportantofthesixconcepts,onethatsupportsalltheothers.Mul-paradigm systems such as Oz provide more choices for notation than sing- paradigm languages.