

1. Record Nr.	UNINA9910144128203321
Titolo	Modular Programming Languages : Joint Modular Languages Conference, JMLC 2000 Zurich, Switzerland, September 6-8, 2000 Proceedings // edited by Jürg Gutknecht, Wolfgang Weck
Pubbl/distr/stampa	Berlin, Heidelberg : , : Springer Berlin Heidelberg : , : Imprint : Springer, , 2000
ISBN	3-540-44519-6
Edizione	[1st ed. 2000.]
Descrizione fisica	1 online resource (X, 302 p.)
Collana	Lecture Notes in Computer Science, , 0302-9743 ; ; 1897
Disciplina	005.13
Soggetti	Programming languages (Electronic computers) Software engineering Computer programming Computer logic Operating systems (Computers) Programming Languages, Compilers, Interpreters Software Engineering/Programming and Operating Systems Software Engineering Programming Techniques Logics and Meanings of Programs 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 at the end of each chapters and index.
Nota di contenuto	The Development of Procedural Programming Languages Personal Contributions and Perspectives -- The Development of Procedural Programming Languages Personal Contributions and Perspectives -- Parallel and Distributed Computing -- Composable Message Semantics in Oberon -- Derivation of Secure Parallel Applications by Means of Module Embedding -- Mianjin: A Parallel Language with a Type System That Governs Global System Behaviour -- A Design Pattern and Programming Framework for Interactive Metacomputing -- Mobile Agents Based on Concurrent Constraint Programming -- Components -- Rethinking Our Trade and Science: From Developing Components to

Component-Based Development -- Explicit Namespaces -- Stand-Alone Messages -- The Design of a COM-Oriented Module System -- Oberon as an Implementation Language for COM Components: A Case Study in Language Interoperability -- Modularisation of Software Configuration Management -- Extensions and Applications -- Leonardo: A Framework for Modeling and Editing Graphical Components -- OMX-FS: An Extended File System Architecture Based on a Generic Object Model -- On Adding a Query Language to a Persistent Object System -- Project C2 – A Survey of an Industrial Embedded Application with Native Oberon for PC -- System Architecture and Design Using Co-operating Groups of Real and Abstract Components -- Compilers and Runtime Environments -- Abstraction and Modularization in the BETA Programming Language -- Design of Multilingual Retargetable Compilers: Experience of the XDS Framework Evolution -- Structuring a Compiler with Active Objects -- A Multiprocessor Kernel for Active Object-Based Systems -- Evaluating the Java Virtual Machine as a Target for Languages Other Than Java -- Building Your Own Tools: An Oberon Industrial Case-Study.

Sommario/riassunto

The circle is closed. The European Modula-2 Conference was originally launched with the goal of increasing the popularity of Modula-2, a programming language created by Niklaus Wirth and his team at ETH Zurich as a successor of Pascal. For more than a decade, the conference has wandered through Europe, passing Bled, Slovenia, in 1987, Loughborough, UK, in 1990, Ulm, Germany, in 1994, and Linz, Austria, in 1997. Now, at the beginning of the new millennium, it is back at its roots in Zurich, Switzerland. While traveling through space and time, the conference has mutated. It has widened its scope and changed its name to Joint Modular Languages Conference (JMLC). With an invariant focus, though, on modular software construction in teaching, research, and “out there” in industry. This topic has never been more important than today, ironically not because of insufficient language support but, quite on the contrary, due to a truly confusing variety of modular concepts offered by modern languages: modules, packages, classes, and components, the newest and still controversial trend. “The recent notion of component is still very vaguely defined, so vaguely, in fact, that it almost seems advisable to ignore it.” (Wirth in his article “Records, Modules, Objects, Classes, Components” in honor of Hoare’s retirement in 1999). Clarification is needed.
