W.M. Seiler
Institut für Algorithmen und Kognitive Systeme, Universität Karlsruhe, D-76128 Karlsruhe, Germany
Zblno.: DE009410313
Author: Andreas Sorgatz
Title: Dynamische Module (Dynamical Modules) (German)
Source: MuPAD Reports, Teubner, Stuttgart 1996
Class.:
Primary Class.: 68N20
Secondary Class.: 68-02, 68Q40
Keywords: dynamical modules, computer algebra systems
Review: Most existing computer algebra systems are designed as closed systems, i.e. the user can only use the functionality provided by the system. However, to solve problems in scientific computing it is often necessary to use a variety of software tools including for example numerical libraries or visualization software. For efficiency reasons it might also be interesting to implement some algebraic algorithms in a language like C. The book of Andreas Sorgatz describes a novel approach for solving this problem by allowing the user to link dynamically, i.e. at run-time, machine code libraries to the kernel of the computer algebra system. This is in marked contrast to other proposed solutions like OpenMath or MathLink which are based on interprocess communication.

The book consists of eight chapters and an appendix. The first chapter gives a general introduction into the problem of communicating with a computer algebra system and discusses advantages and disadvantages of some possible solutions. Chapters 2 and 3 describe the basic concept of a dynamical module and its implementation in the computer algebra system MuPAD. After a discussion of some design criteria a detailed six layer model is developed which forms the basis of the implementation. The next two chapters deal with the problem of generating dynamical modules. Chapter 4 describes *mmg*, the MuPAD Module Generator, a program for compiling and linking C/C++ programs to dynamical modules. In order to simplify the development of modules a toolbox was implemented with many predefined types and procedures. Its description in Chapter 5 emphasizes especially the problem of accessing and manipulating MuPAD objects within a dynamical module. Chapters 6 and 7 present two advanced applications: a C-caller version of MuPAD and a compiler for MuPAD code. The latter one transforms MuPAD code, i.e. procedures written in the high level language provided by MuPAD into the C code that can be handled by the module generator. The last chapter gives a brief outlook on how the author sees the future development of his implementation. The appendix contains some concrete examples of dynamical modules and brief descriptions of some functions for the administration of dynamical modules within MuPAD.

The book is a technical description of the concrete realization of dynamical modules within MuPAD. It is not intended as a manual for MuPAD users who want to develop their own dynamical modules, although the included examples may help a bit. The main audience of this book are developers of computer algebra systems or users who want to know precisely what happens internally.
Remarks: