Astronomical Data Analysis Software and Systems VIII
ASP Conference Series, Vol. 172, 1999
D. M. Mehringer, R. L. Plante, and D. A. Roberts, eds.

Automated Programming for Physics and Astrophysics

Michele Maris

Astronomical Observatory of Trieste (OAT), Via G.B. Tiepolo 11, I-34131, Trieste, Italy

Maria Staniszkis

Mathematics and Informatics Department, Udine University, Via delle Scienze 206, I-33100, Udine, Italy

Abstract. The present communication describes the effort to develop a model of Automated Program Generator (APG) proposed for the production of the simulation software for the ESA Planck/LFI mission.

1. Introduction

Considering that the core of a simulation project is the production of the simulation software, the problems related to its creation, development and documentation represents a major part of the whole work. In order to facilitate these tasks, in the past years many tools have been proposed (Cook, Painter, & Brown 1991; Cook & Painter 1992; Brown 1992; Dubois 1994; Dubois & Yang 1996; Beazley & Lomdahl 1997), but from the user's point of view the ideal solution would be one in which the computer is given an abstract description of the required simulation and is left to the task of producing the code. This is possible if most operations required to support and execute a simulation could be standardized, allowing the related code to be specified by a high level language; this leads us to *Automatic Programming* and *Program Generation* concepts. A sort of "software robot" called the *Automated Program Generator* (*APG*) is presented, which generates simulation software starting from a specification given by the user. Moreover, if we suppose that the simulations are partitioned into *simulation classes*, it is possible to build a library of standard reusable simulations.

The approach to automatic programming presented here was first proposed for the Planck/LFI collaboration (Mandolesi et al. 1998) where it was applied to explore solutions to the on-board data compression problem (Maris et al. 1999), but in its simplest form it was also applied to solar neutrino (astro)physics giving excellent results (Maris 1999).

2. A Simulation Model

The design of an APG system has to be founded on a *model of scientific simulations* whose role is to define both the basic components of a simulation project and their purposes and relations in the management of the simulation cycle.

Non-interactive scientific simulations are considered in the present communication: the *simulator* receives as input a sequence of simulations to be executed and, without any interaction with the user, produces as output numerical results in forms of tables, data files, or graphs. The real world *experiments* may use the same or slightly modified *experimental apparatus* and varying *experimental conditions* often in dependence from obtained *results*. Considering a simulator run a *computer world experiment*, we can view the *input parameters* as experimental conditions and an implementation of *simulation functions* and *output forms* as an experimental apparatus. Furthermore, we can consider a *computer experiment* fully documented when all its parts are described in the relative documentation.

The components of a simulation model are: the *simulation parameters space* – the set of parameters which guide the simulator behavior; *the simulation results space* – the values of the simulation outputs; the *parameters hierarchies* which order parameters and results; the *sampling mesh for parameters* defined as the set of discrete sampling functions; *the constraints over parameters* which define the computation regions in a mesh and at last the *parameters interdependencies*. A simulation, then, may be seen as a process applied to parameters producing requested results and controlled by the exploration strategy defined by the sampling mesh.

3. The APG

The APG is supposed to perform the *automatic program generation* from high level specifications supplied by the *user* and/or by a *database*. Compiling them (with the aid of external compilers and linkers when required), it assembles an executable code including specialized functions and building blocks. The produced code should satisfy the following requirements: is specialized for a given scope and independent from hardware and software platforms, merges many software packages and libraries in an hybrid architecture through *cascades of script files*, and is self–documented.

In order to produce a useful and working tool, a risk assessment analysis was conducted considering three risk areas: Users, Technology, Design. The resulting constraints led to the definition of a user requirement document and of the implementation strategy. Between the basic elements of the strategy the most remarkable ones are: the design of a specialized specification language for simulations and the use of the *multi-language strategy* which means that tasks may be described in different programming languages.

In summary an APG for simulation purposes should be a software which, given a high level simulation specification, hierarchies and constraints on parameters, is able to construct the parameter space and to generate in a consistent order the requested computations. In particular the process of the mesh generation and its exploration can be seen as a kind of *intelligent scanning process* of the parameters space. The proposed APG system is mainly composed by the following elements:

Specification Language which allows the user to describe a specific simulation through a specification document;

APG Engine which combines the information from the specification document with that from the database to obtain the low level code.

Data Base which stores information about: procedures, functions, methods and modules, software documentation, previous simulations;

Simulations Data Base which stores information about: starting data, intermediate and final results, log files and simulations documentation;

Execution Manager which supervises the execution of the generated simulation;

Report Generator which generates simulation reports;

Error and Messages Manager reacts to errors, exceptions, traps and messages between modules.

4. Simulation Specification

Assuming that the specification language should be easily understood and used by a scientist whose main concern is the study of a physical process reproduced by the simulation and its numerical convergence, the developers of the proposed APG wish to design an expressive specification language and at the same time to take care of the programming.

The specifications of simulations in the APG under construction are divided into three parts:

1. definitions of parameters and functions. In the initial version the parameters could be specified by a single value, or by a list of values, a list of expressions containing loops and application of functions which could be fetched from the data bases, a reference to a file (typically for parameters obtained as output of a previous simulation), another simulation. All the functions and transformations could be defined in any of the foreseen languages or through references to data bases, where the APG compiler is going to fetch them;
2. optional definition of an order in which the parameters contribute to the mesh construction;
3. the final part, where it is possible to indicate the functions for the simulation, presents the possibility of the incremental design of simulations.

At this point the compiler produces the actual code of the simulator, by ordering the code for the "intelligent" scanning of the mesh and then by calling the simulation functions according to parameters dependencies and hierarchies.

5. Conclusions

The principles of an Automated Program Generator (APG) system useful to manage simulations at a high level of abstraction were discussed in their various aspects. The realization of such a system requires the development of a specific language, whose design has to be based on a model of the tasks commonly accomplished by simulations in both physics and astrophysics. The language and its compiler is presently being developed and a beta release is expected in a short time.

The first application planned for such system will be the management of the simulations needed by the microwave Low Frequency Instrument consortium for the European Space Agency (ESA) Planck mission.

Acknowledgments. Many people supported the effort of the APG development and deserve acknowledgments. In particular Dr. M. Pucillo, Dr. F. Pasian, Dr. R. Smareglia and Dr. M. Callegari of OAT. M. M. is indebted scientifically with J.F. Painter of Lawrence Livermore Laboratories at UCLA and with Dr. G. Maris, Dr. A. Bignoli, Dr. G. Bollani and Mr L. Terzaroli of the Research and Development Division of Etnoteam, Milan.

References

Beazley, D. M. & Lomdahl, P. S. 1997, Comp. in Phys., 11, 230

Brown, S. A. 1992, Comp. in Phys., 6, 262

Cook, G. O., Jr. & Painter, J. F. 1992, in Expert Systems for Scientific Computing, ed. E. N. Houstis, J. R. Rice, & R. Vichnevestsky (North-Holland: Elsevier), 401

Cook, G. O., Jr., Painter, J. F., & Brown, S. A. 1991, J. Sci. Comp., 6, 193

Dubois, P. F. 1994, Comp. in Phys., 8, 70

Dubois, P. F. & Yang, T.-Y. 1996, Comp. in Phys., 10, 359

Mandolesi N., et al. 1998, Low Frequency Instrument for Planck, Proposal submitted to ESA

Maris, M. 1999, this volume, 50

Maris, M., Pasian, F., Smareglia, R., Maino, D., Burigana, C., Staniszkis, M., & Barriga, J. 1999, this volume, 145