

**UniVista: A Universal Graphical User Interface for
Scientific Modeling Codes
Status Report
June 14, 1999
Daren Stotler & Russell Hulse**

1 Introduction

The increasing complexity of scientific modeling codes has made collaborative code usage and development an increasingly critical issue. In addition, decreasing budgets in the field of magnetic confinement fusion have also added to the need for researchers to efficiently share simulation codes.

Most such codes are written in FORTRAN, have primitive user interfaces, and little or no documentation. The code author is faced with choosing one of three alternatives when he decides to pass his code along to new users:

1. **Do nothing.** He can leave the new users to struggle through the code and documentation. At best, this leads to significant wasted effort on the part of the user. At worst, it can in addition lead to significant opportunities for error, or even complete loss of the scientific investment in the code if the effort to re-use the code proves too difficult and is abandoned.
2. **Help users as needed.** The author can walk a new user through the code to get him started and then follow-up by answering questions over the phone or via E-mail. However, the code author will typically find himself spending a significant fraction of his time carrying out this tutorial and answering routine questions, rather than discussing real physics issues.
3. **Make the code easier to use.** The chief reason for not choosing this option is that it requires an up-front investment in time and, perhaps, money. Both being in short supply, options (1) and (2) above, although clearly undesirable, are frequently selected by default. Over the long run, however, an adroit investment in developing high-level interface technology is a much more effective and efficient approach.

A variety of modern software tools exist to help the researcher repackage his legacy FORTRAN code with a more user-friendly interface. In most cases,

though, he must start from scratch and scale the learning curve of a sophisticated integrated development environment or programming language. The overwhelming preference for the first two of the above choices arises from the size of these learning curves. Furthermore, the resulting code interface may be difficult to modify, discouraging the code author from keeping it up-to-date as the code's input variables are altered.

What is needed instead is a tool specifically designed for this task: a simple framework which will allow a researcher to quickly design and customize a graphical interface for his legacy FORTRAN code. The tool should make documentation of the code and its input variables accessible to the end user through this graphical interface. The tool should be sufficiently simple to use that the code author will be motivated to create multiple graphical interfaces based on a common description of his code, each tailored to different user requirements. For example, there can be separate interfaces suitable for users of different levels of expertise, and/or interfaces customized via specialized defaults and user input presentations as required to set up detailed investigations of various physical scenarios.

UniVista is that tool. The design objectives of UniVista are discussed in more detail in the appended report.

2 History

Russell Hulse and Walt Stark made a first attempt at building UniVista with Java in 1996. However, the immaturity of the language repeatedly caused problems for Stark. Although he did make considerable progress, UniVista was not suitable for practical use when he left PPPL in 1997. Budget constraints had forced Stark to do the UniVista development work on his personal PC. Consequently, the code was essentially lost once Stark resigned, and the project was tabled indefinitely.

At the same time, work on the DEGAS 2 Monte Carlo neutral transport code was progressing. Its principal authors, Charles Karney and Daren Stotler, had sought to make DEGAS 2 as user-friendly and widely used as its predecessor, DEGAS. However, DEGAS 2 was also designed to be very flexible. Since a more flexible code is generally more complicated than one which is more specialized, the authors were aware that considerable effort would be required in designing the user interface for DEGAS 2.

Distribution of DEGAS 2 to other users began in early 1999 once the code could be productively coupled to a plasma transport code. At this stage, though, the code still possessed a primitive user interface. The increasingly urgent need to

develop a better interface led to the resurrection of UniVista.

In a limited-resource environment, the key to making efficient progress is to endow new ventures with both near-term, tactical value as well as long-term, or strategic benefits. In this case, UniVista would address the tactical problem by providing a coherent, easy-to-use interface to DEGAS 2. The strategic value would be the production of a prototype application of UniVista. Demonstrations of this prototype would convince other code authors of the efficacy of the UniVista approach. Subsequent application of UniVista to those other codes would be straightforward.

3 A Student Project

Russell Hulse and Daren Stotler advertised the construction of UniVista as a topic for a Princeton University Computer Science undergraduate special projects course through Prof. J. P. Singh. By going this route, three desirable objectives were achieved:

1. The person brought in to work on UniVista would likely be familiar with the requisite, cutting-edge technologies.
2. The work would be done at little or no cost to PPPL.
3. An additional mode of collaboration building mutually valuable contacts between PPPL and the Computer Science Department would be established.

In January 1999, Computer Science / Physics major Edward Peterlin expressed his desire to work on UniVista. The first month of the project was spent laying out the design specifications of UniVista in detail. Peterlin's intimate familiarity with Java (including differences between recent versions) and database tools was very helpful in this process. Peterlin spent the rest of the semester writing UniVista itself. Weekly meetings with Hulse and Stotler tracked his progress. The status of the project at the end of the course was detailed in the appended report submitted by Peterlin to the Computer Science Department.

Although Peterlin had been extremely productive, UniVista still needed additional work to before it could be used for its intended purpose. Prompted by a memo from Andrea Moten advertising the cost-effectiveness of University summer students, Hulse and Stotler sought to hire Peterlin full-time for the month between completion of his coursework and departure for his new career in California. Peterlin gave a seminar and demonstration on UniVista to the CPPG group

midway through the period. The end result of Peterlin's effort was a "proof-of-principle" demonstration: the UniVista design objectives could indeed be met with existing Java and database tools.

4 Applications

The proof-of-principle success of UniVista is very encouraging, as is the effectiveness of harnessing bright Princeton University computer science undergraduates to develop valuable computing tools such as UniVista tailored to the laboratory's needs. *The value PPPL received in return for less than one month of Peterlin's salary is difficult to overstate.* All told, UniVista represents 12,000 lines of well-designed, object-oriented, internally documented code. Just about any other PPPL employee would be hard pressed to duplicate this effort.

As a result of Peterlin's efforts, UniVista is now very close to being useful for practical work. In addition to the DEGAS 2 application, Alexander Pletzer has also expressed an interest in UniVista. Several others are waiting in line. However, certain refinements and adaptations to meet specific user requests are needed before the code is handed over to its intended customers. Also, as with any new software, one key to successful adoption is some level of continued development effort and extension of code functionality in response to user feedback. Some of these suggested extensions are quite straightforward, due to the object-oriented design of the UniVista system. Others are sufficiently ambitious that they could be the basis of additional student undergraduate special project courses.

5 Proposed Action

UniVista proof-of-principle has been established. UniVista's application to selected PPPL codes should now be supported through continued student work, either as part-time PPPL employees, and/or via special projects courses.

Appendix

The report submitted by Peterlin to the Computer Science Department is attached. Additional information on UniVista can be found on the World Wide Web at:
<http://w3.pppl.gov/~dstotler/UniVista/>.