VME: A VISUAL MODEL EDITOR FOR THE ANSWERS SHIELDING AND CRITICALITY CODES

David DEWAR, Andrew J COOPER, Paul HULSE

H460, British Nuclear Group,

Risley, Warrington, Cheshire, WA3 6QH, UK

and

Pat COWAN

Serco Assurance,

Winfrith, Dorchester, Dorset, DT2 8DH, UK

ABSTRACT

Criticality and radiation shielding analysts in the British Nuclear Group use codes produced by the Nuclear Codes Development (NCD) consortium between British Nuclear Group and Serco Assurance (marketed via the ANSWERS Software Service) in the development of their nuclear safety cases. These computational codes process models created by the analysts. The models use Fractal Geometry, a form of Constructive Solid Geometry, and Hole Geometry (a method of modelling complex shapes in a very efficient way). In order to successfully produce a model that when executed will produce useful results, an analyst must undertake the 'build-check-edit cycle' with the model. The 'build' and 'edit' of a model is an exercise undertaken by hand. The VISTA suite of codes produced by ANSWERS supports the 'check' but the model definition remains primarily a textual exercise. Users now expect a graphical interface to any task and the Visual Model Editor (VME) has been designed with this in mind.

VME has been developed, as part of the NCD consortium, to enable users to more easily create, check and submit models for execution. VME allows users to build and edit models interactively with a 3D display and edit dialogs. VME can show models successfully both as a wireframe display and as ray-traced slices.

In this paper, we present the design of VME and how it can be used to create and check a model. We also show how VME interacts with the ANSWERS code LAUNCHPAD in order to submit a job, and how VME can be used for interactive error checking and visual post-processing.

VME is written in Java and we demonstrate how the portability and common look-and-feel that Java provides enhance the productivity gains that VME gives.

In addition, in this paper, we introduce some of the new capabilities to be introduced into VME, including selective display from a parameterised model, and enhancements to include use of a native OpenGL library.

Keywords: Visual Model Editor, VME, ANSWERS codes

1. INTRODUCTION

VME works with models for the ANSWERS criticality code MONK [1] and the ANSWERS shielding codes MCBEND [2] and RANKERN [3].

MONK is a program that employs the Monte Carlo method to study criticality safety and reactor physics problems. The primary aim of MONK is to calculate the neutron multiplication factor (k-effective) of a model by the simulation of the birth, migration and ultimate fate of a finite number of neutrons.

MCBEND is a radiation transport program that is used extensively in the UK nuclear industry to predict radiation levels for new and existing plant. MCBEND can calculate the neutron, gamma or electron flux at any point in a computational model. MCBEND uses the Monte Carlo method to produce its predictions.

RANKERN is a program for the design and assessment of gamma-ray shielding. It employs the integrated point-kernel technique in its calculations.

MONK, MCBEND and RANKERN use the Fractal Geometry (FG) modelling method [4]. FG is a Constructive Solid Geometry (CSG) representation system especially developed for these codes. FG comprises a solid-body package that uses a simple set of bodies, including the sphere, box, rod, prism, cone and torus. These bodies possess a general orientation and can overlap, if necessary. From this construction, a 'part' is created that possesses its own local coordinate system. The parts are then included in bodies in other parts to form the whole. In addition, the codes MONK and MCBEND employ the Hole Geometry (HG) modelling method [5] that utilises a tracking algorithm that does not

rely on element boundaries. The hole geometries rely on a technique called Woodcock tracking, and comprise a set of hole types that can be included inside FG bodies or other holes. This geometry enables more complicated shapes to be modelled than would otherwise be possible.

The criticality and shielding analysts use technical drawings and other technical information for the creation of their models. The creation of these models is currently a textual exercise. The models are then verified. The model verification consists of two aspects : checking that the model accurately reflects the underlying design; and that it is a well-formed model that a computational code can successfully execute.

The models are checked with the assistance of the VISTA suite of visualisation codes produced by ANSWERS and then edited to correct any errors. This is known as the 'build-check-edit' cycle.

The VISTA suite includes the programs VISTA-WIRE [6], VISTA-RAY [7] and VISAGE [8].

VISTA-WIRE produces 3D images of a model using the Programmer's Hierarchical Interactive Graphics System (PHIGS) [9]. It generates wireframe images to allow the rapid traversal of a model, and allows the display of the model's geometrical bodies, source bodies, dose points, splitting meshes and scatter/reflect bodies.

VISTA-RAY produces 3D images of a model using simple ray-tracing techniques. The geometry of the model can be displayed in terms of its component materials, zones or regions, with the options to display dose points, source bodies and splitting meshes.

VISAGE is a high-resolution interactive graphics tool for the display and verification of a model. It produces slices through a model using ray-tracing techniques. The geometry of the model can be displayed in terms of its component materials, zones or regions. VISAGE also gives the options to display source bodies, splitting mesh, dose points and scatter/reflect bodies.

2. PROBLEMS WITH THE CURRENT APPROACH

VME is designed to increase the productivity of the criticality and shielding analyst. The economic realities of UK nuclear industry today imply that companies bidding for work from the Nuclear Decommissioning Agency (NDA) will remain under significant pressure to reduce the cost of their work.

Manual production of a model is time-consuming, particularly the iterative loop of design and verification, and potentially error-ridden. Each code of the VISTA suite performs a subset of the tasks required during model verification. This can require the user to flip between the different codes complicating the process.

The checking process is further complicated by the requirement to inspect output listings of executed jobs. This must be done either by hand or with a text editor.

The current approach to model building and editing is primarily textual. The increasing use of graphical-based computer systems means that this is what the users expect. This can be seen with MCNP and its Visual Editor (MCNP-VISED) [10], and its visualisation tools : MORITZ [11] that allows dynamic 3D visualisation coupled with the ability to define and edit geometry; and SABRINA [12] for 3D geometry visualisation using ray-tracing. The codes of the VISTA suite do not provide any direct method of editing the model code, and whilst they are in themselves valuable visualisation tools, they still force the check part of the 'build-check-edit' cycle to be distinct from the 'build' and 'edit' parts.

The codes of the VISTA suite use a variety of methods for their graphical display. VISAGE and VISTA-RAY employ calls to the Motif and Xlib libraries for 2D and 3D plotting in their respective cases. VISTA-WIRE uses the GPHIGS graphics package [13] (a commercial cross-platform implementation of the PHIGS+ standard) for 3D plotting. The use of these different methods means that the codes must be ported for each platform in order to use operating-system dependent libraries, and means on systems using Microsoft Windows that the codes rely on the presence of an additional X-server application.

3. DESIGN OF VME

VME is designed to be a 'one-stop shop' where users can edit, verify, then execute and post-process their models. It is designed to use a Microsoft Windows look-and-feel that users are familiar with so reducing the demands for user training.

VME is designed to read all current MONK, MCBEND and RANKERN models.

VME provides a set of model display options. The user can choose between either a set of 3 orthogonal 2D displays with an isometric 3D display or a fully-interactive 3D view. The display gives a choice of parallel and perspective projection, and allows the user to display wireframe, solid views and ray-traced slices. The use of a wireframe allows a model to be quickly traversed before slices, that have a longer image generation time, are created.

Ray-tracing uses SKETCH [14], the drawing engine for the VISTA suite of codes. SKETCH uses the same code to parse a model as the computational codes. The use of SKETCH also gives VME the ability to detect for multiple zone definitions which is a common error introduced during model creation.

VME includes a tree-view that gives an exploded view of a model and its component units, for example geometry specification, material specification, source geometry

specification etc.. The tree-view offers the user the ability to invoke dialog boxes that allow the text of an individual unit to be edited. The displayed text is colour-coded for comments and keywords to enhance the clarity of the displayed model text.

VME also provides an edit window that allows the complete textual model to be edited and re-parsed by VME. This ability to edit either component units or the whole model allows both novice and experienced users to interact with the model at a level they are comfortable VME allows the user to invoke a dialog box containing the output listing for the executed run for a model. VME colour codes any errors in this listing and provides links to the pertinent input units so that a dialog box containing the text for that unit is invoked if the error is selected.

VME also allows the ability to display chosen output values in a graphical form, for instance as a graph.

4. VME : A PRODUCTIVE CROSS-PLATFORM APPLICATION

The ANSWERS codes are designed to run on all the common platforms, including SUN Solaris, Windows NT/XP, and Red Hat Linux.

VME has been designed to be readily portable across platforms and to provide a common look-and-feel on all platforms. To this end, VME has been written in the cross-platform development language Java [15] with use of the Java Swing Graphical User Interface (GUI) library to create the interface.

VME uses an open-source Java OpenGL-emulation package, jGL [16], to implement its fully-interactive 3D view. JGL is a set of Java classes that provide an API similar to that of OpenGL but uses standard Java library drawing routines at its lower level. The use of this emulation ensures that there is no call to any underlying system libraries and so preserves the cross-platform nature of the VME release.

The use of an OpenGL-compliant API, an API that a large pool of programmers are familiar with, in conjunction with the very popular Java development language, guarantees that programmer productivity is increased and sustainable.

The use of open-source applications to provide functionality for VME has also increased the development speed. An example of this is the use of an open-source package for the graphical post-processing. This package, JFreeChart [17], is a Java class library for generating charts and other graphics displays.

5. FURTHER WORK

Further work to VME will centre around ensuring its functionality and appearance match user expectations, expanding the aspects of the model it can display, and speeding-up the interactive display response.

with. Similarly to assist novice users, dialog boxes can be optionally displayed that show, in a formatted fashion, the material and zone specifications.

VME has a link to LAUNCHPAD [18], the calculation management utility produced by ANSWERS. This allows the user to create a job submission deck for a model and to submit the job for execution.

VME couples the ability to execute jobs with the ability to syntactically analyse and post-process the results.

The models for the ANSWERS codes can be parameterised. At present, VME replaces all array elements by the first indexed value and all variables by their ascribed or calculated values. Further development of VME is to occur to allow selective display of the individual input for a set of parameters.

With respect to the fully-interactive 3D display, jGL provides all the functionality currently required from VME. However, doubts have since arisen about its vitality as an open-source project so there are concerns about relying on it as a key component of VME.

In addition, if VME is developed to use a native OpenGL implementation, programmers will still be working with the well-understood API but VME will be able to exploit such native features as hardware acceleration.

One possible native implementation is JOGL [19]. JOGL is a reference implementation of the Java bindings for OpenGL sponsored by SUN Microsystems.

In addition, work has been undertaken to investigate the use of the graphical toolkit VTK [20].

6. REFERENCES

[1] SERCO Assurance, "**MONK – A Monte Carlo Program for Nuclear Criticality Safety and Reactor Physics Analyses. User Guide for Version 8**", ANSWERS/MONK(98)6.

[2] SERCO ASSURANCE, "**MCBEND – A Monte Carlo Program for General Radiation Transport Solutions. User Guide for Version 10**", ANSWERS/MCBEND/REPORT/004.

[3] SERCO Assurance, "**RANKERN – A Point Kernel Program for Gamma-Ray Transport Solutions. User Guide for Version 14**", ANSWERS/RANKERN(95)03.

[4] N Smith et al, "Geometry Modelling and Visualisation for the Monte Carlo Code MCBEND", 8th International Conference on Radiation Shielding, April 1994, Arlington Texas USA.

[5] N R Smith and G A Morrell, "An Introduction to MONK 7", 5th International Conference on Nuclear Criticality Safety (ICNC 95), September 1995, Albuquerque New Mexico USA.

[6] SERCO Assurance, "**VISTA-WIRE – A Wire-Frame Geometry Visualisation Package for MCBEND, RANKERN and MONK. User Guide for Version 2**", ANSWERS/VISTA(98)8.

[7] SERCO Assurance, "**VISTA-RAY – A Ray-tracing Geometry Visualisation Package for MONK, MCBEND and RANKERN. User Guide for Version 3A**", ANSWERS/VISTA/REPORT/003.

[8] SERCO Assurance, "**VISAGE – A Program for the Graphical Validation of MONK,MCBEND and RANKERN. User Guide for Version 5A**", ANSWERS/VISAGE/REPORT/002.

[9] **PHIGS Standard**, www.itl.nist.gov/iaui/vvrg/cugini/pvt/hy-std.html .

[10] **Visual Editor Consultants**, www.mcnpvised.com .

[11] **MORITZ**, www.whiterockscience.com/wrs.html .

[12] **SABRINA**, www.whiterockscience.com/sabrina/sabrina.html .

[13] Mercury Computer Systems Inc., **GPHIGS+ Overview**, www.tgs.com/index.htm?pro_div/phigs_main.htm~main .

[14] SERCO Assurance. "**SKETCH – A Program for Checking Geometry Models used in MONK, MCBEND and RANKERN. User Guide for Version 2D"**, ANSWERS/SKETC/REPORT/001.

[15] **Java Technology**, java.sun.com

[16} **jGL**, **3D Graphics Library for Java**, www.cmlab.csie.ntu.edu.tw/~robin/JavaGL

[17] **JFreeChart**, www.jfree.org/jfreechart/ .

[18] SERCO Assurance, "**LAUNCHPAD – A Calculation Management Utility for ANSWERS Software, User Guide**, **Issue 4**", ANSWERS/LAUNCHPAD/REPORT/001.

[19] **JOGL Project**, jogl.dev.java.net .

[20] **VTK, The Visualisation Toolkit**, public.kitware.com/VTK/index.php .