5. Методические указания по созданию постоянно действующих геолого-технологических моделей нефтяных и газонефтяных месторождений (Часть 2. Фильтрационные модели). Утв. Заместителем Министра энергетики РФ Шелеповым В.В. 22.01.2002 г. М.: ОАО «ВНИИОЭНГ», 2003.

6. Zgangxin Chen, Gyamrem Huan, Yuanle Ma Computational methods for multiphase flows in porous media. Dallas, Texas: Southern Methodist University, 2006.

7. Peaceman D. W. Interpretitation of well-block pressure in numerical reservoir simulation // SPE Journal. 1983. Vol. 23, N 3. P. 531–543.

8. Сидоров М. Л., Пронин В. А. Неструктурированная призматическая дискретизация сложных геологических структур в параллельном режиме // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2015.

9. Бартенев Ю. Г., Капорин И. Е., Харченко С. А., Сысоев А. В. и др. Комплекс библиотек параллельных решателей СЛАУ LParSol версии 3 // XIV Международная конференция «Высокопроизводительные параллельные вычисления на кластерных системах (HPC 2014)»: сб. трудов. Пермь, 10–12 ноября 2014. С. 49–53.

10. Nghiem L. S., Collins D. A., Sharma R., Seventh SPE comparative solution project: Modeling of horizontal wells in reservoir simulation, SPE 21221 // The 11th SPE Symposium on Reservoir Simulation, Anaheim, CA. 1991.

11. Douglas J. Jr., Peaceman D. W., Rachford H. H. Jr. A method for calculating multi-dimensional immiscible displacement // Trans. SPE AIME 1959. 216. 297–306.

ПАРАЛЛЕЛЬНАЯ СИСТЕМА ПОСТОБРАБОТКИ SCIENTIFICVIEW. МАКРОЯЗЫК

А. И. Лопаткин, А. Л. Потехин, В. В. Жирнов, Д. С. Кондратьев, В. В. Ломтев, Е. В. Нестеров, П. А. Тюхтина, Ю. В. Козачек, А. К. Меньшикова, И. В. Логинов

Российский Федеральный Ядерный Центр –
Всероссийский НИИ экспериментальной физики, г. Саров

Процесс обработки данных с использованием систем постобработки и визуализации зачастую может представлять последовательность однотипных действий, которые необходимо выполнить пользователю через графический интерфейс приложения. С целью автоматизации обработки данных современные системы визуализации и постобработки предоставляют возможность управления приложением с помощью команд макроязыка. Макроязык представляет собой набор команд, посредством которых пользователь имеет возможность выполнять действия, доступные через графический интерфейс пользователя. Макроязык позволяет с помощью небольших программ создавать сложные процедуры обработки, для выполнения которых через графический интерфейс потребовались бы часы или даже сутки работы пользователя. Кроме того, макропрограммы можно применять к разным результатам моделирования, создавая типовые процедуры обработки для определенного класса задач.

Для автоматизации обработки данных в параллельной системе постобработки ScientficView [1–2] реализована возможность управления системой, используя команды разработанного макроязыка.

Система ScientificView предназначена для фильтрации, отображения, числового анализа результатов моделирования физических процессов на сетках регулярного и нерегулярного типа, а также для обработки данных, полученных бессеточными методами моделирования (частицы, молекулы, кластеры).

Для работы с макроязыком в рамках ScientificView, начиная с версии 1.8.0.0, добавлен специальный интерфейс пользователя. Использован интерфейс для работы с макроязыком препроцессора комплекса программ ЛОГОС. Данный интерфейс предоставляет два режима работы с командами макроязыка: режим интерактивной консоли и режим редактора кода. Внешний вид диалога для работы в режиме интерактивной консоли приведен на рис. 1.

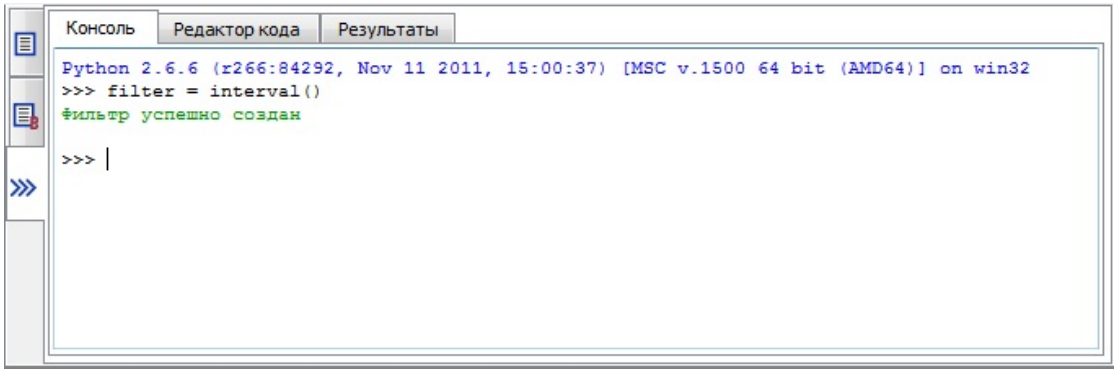

Рис. 1. Интерфейс работы с макроязыком в режиме интерактивной консоли

В режиме интерактивной консоли выполнение введенных пользователем команд происходит сразу же после их ввода. Интерфейс редактора кода приведен на рис. 2.

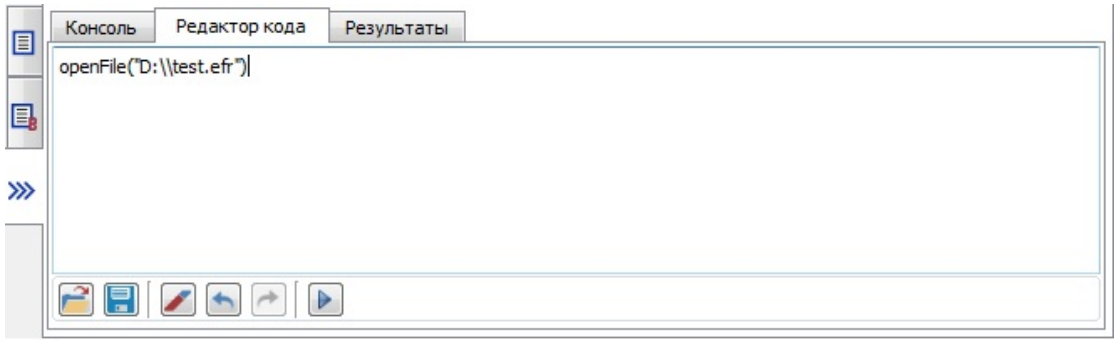

Рис. 2. Интерфейс работы с макроязыком в режиме интерактивной консоли

В режиме редактора кода пользователь может вводить несколько команд, формируя из них программы, редактировать введенные команды и запускать сформированные макропрограммы на выполнение. Также редактор позволяет загрузить ранее написанную пользователем программу или сохранить текущую.

В качестве основы разрабатываемого в системе ScientificView макроязыка использован язык Python версии 2.6. Соответственно при написании макропрограмм допускается использование всех возможностей данного языка. Для реализации макрокоманд использовался Python/C API, который позволяет создавать модули расширения для языка Python на основе кода на языках C/C++.

На текущий момент макроязык системы ScientificView насчитывает порядка 60 команд, охватывающих наиболее значимые возможности системы. Реализованные команды условно можно разделить на следующие группы:

- работа с файлами (открытие файла с сеточными данными, запись данных в файл);
- команды создания алгоритмов фильтрации;
- команды для изменения параметров алгоритмов фильтрации;
- команды для работы с временными шагами;
- команды интерполяции и калькуляции сеточных массивов;
- команды поиска интегральных характеристик;

- команды создания презентационных материалов;
- команды для подключения к удаленной машине;
- прочие команды.

Преимущественно все макрокоманды с точки зрения пользователя оформлены в виде отдельных функций, то есть основные операции, доступные через графический интерфейс пользователя системы, представлены в виде отдельных функций-команд. Так, например, команда открытия файла выглядит следующим образом:

openFile(«Имя файла»).

Объектно-ориентированный подход используется для управления алгоритмами фильтрации и исходными данными системы ScientificView. Для представления алгоритмов фильтрации в рамках макроязыка реализован отдельный класс. Данный класс содержит информацию о типе алгоритма фильтрации, а также порядковый номер элемента в контейнере элементов сцены и предоставляет следующие методы для установки и изменения параметров алгоритмов фильтрации:

- *setParams(«Набор параметров»)* – набор параметров является уникальным для каждого алгоритма. Функция позволят установить сразу все параметры фильтра;
- *set(«Имя параметра», Значение параметра)*, где «Имя параметра» – имя параметра, значение которого необходимо изменить. «Значение параметра» – устанавливаемое значение. Данную функцию имеет смысл использовать, только если необходимо изменить какой-то конкретный параметр алгоритма. Каждый алгоритм имеет собственное именование параметров, однако, есть и общие для ряда алгоритмов параметры;
- *build()* – запуск процедуры построения фильтра;
- *showArray(«Имя массива для отображения»)* – устанавливает указанный массив для отображения;
- *setOpacity(Значение прозрачности)* – устанавливается степень прозрачности для текущего фильтра. Значение устанавливается от 0 до 1;
- *saveGraph(«Имя файла для сохранения»)* – функция работает для алгоритмов, результатом которых являются графики. Функция предназначена для сохранения полученных графиков в файл, в качестве параметра передается имя файла для сохранения;
- *setShowParam(«Имя параметра», Значения параметра)* – функция предназначена для установки параметров отображения фильтра или элемента с исходными данными. В качестве имени параметра могут выступать следующие строковые константы:
  - «grid» – включить/выключить отображение сетки,
  - «fill» – включить/выключить отображение полигонов,
  - «point» – включить/выключить отображение узлов сетки,
  - «gabarits» – включить/выключить отображение габаритов,
  - «carcass» – включить/выключить отображение каркаса,
  - «text» – включить/выключить отображение текста для ячеек сетки;
- setVisibility(True/False) – установка признака отображения для элемента сцены.

Применение же самого алгоритма фильтрации через команды макроязыка может быть выполнено двумя способами:

- командой *createFilter(«Тип алгоритма фильтрации»),* где параметр «Тип алгоритма фильтрации» принимает одно из предопределенных значений. Например, для применения алгоритма фильтрации «Интервал», предназначенного для выделения части ячеек, значения сеточной величины в которых соответствуют пороговым значениям, используется команда *filter = createFilter(INTERVAL)*;
- используя реализованную для каждого алгоритма команду. Например, создание того же фильтра «Интервал» выглядит следующим образом

filter = interval().

Результатом работы этих команд является объект, с которым возможна дальнейшая работа. Далее пользователю необходимо задать интересующие его параметры алгоритма фильтрации и запустить команду выполнения фильтра. Для возможности применения иерархической фильтрации,

когда к результатам работы одного алгоритма можно применить другой алгоритм фильтрации, в качестве дополнительного параметра в функцию createFilter передается либо имя элемента сцены, к которому необходимо применить новый алгоритм фильтрации, либо непосредственно объект ранее созданного фильтра.

Далее в качестве примера макропрограммы приведено задание параметров алгоритма фильтрации «Интервал».

```
filter = createFilter(INTERVAL)
filter.setParams(«Material», 0, 5)
filter.build()
```

В данном примере функция *setParams* устанавливает параметры фильтра, указывая, что в результат алгоритма должны попасть ячейки, значение сеточной величины с именем *«Material»* в диапазоне от 0 до 5.

В рамках работы, описанной в докладе, обеспечена возможность обработки данных в параллельной системе постобработки ScientificView, не только используя графический интерфейс пользователя системы, но и посредством команд разработанного макроязыка. При этом использование команд макроязыка зачастую позволяет автоматизировать процесс обработки данных и существенно сэкономить время пользователя.

Литература

1. Потехин А. Л., Логинов И. В., Козачек Ю. В. и др. ScientificView – параллельная система постобработки результатов, полученных при численном моделировании физических процессов // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2008. Вып. 4. С. 37–45.

2. Потехин А. Л., Логинов И. В., Тарасов В. И. и др. ScientificView – параллельная система постобработки результатов, полученных при численном моделировании физических процессов // XVIII Международная конференция по компьютерной графике и зрению «Графикон»: сб. трудов. Москва, 23–27 июня 2008. С. 192–198.

РАЗВИТИЕ ПРОГРАММЫ *LUCKY* ДЛЯ РЕШЕНИЯ УСЛОВНО-КРИТИЧЕСКИХ И НЕСТАЦИОНАРНЫХ ЗАДАЧ НЕЙТРОННОЙ ФИЗИКИ

А. В. Моряков

Национальный исследовательский центр «Курчатовский институт», г. Москва

Введение

Основными элементами современной вычислительной техники являются суперкомпьютеры – вычислительные машины с большим числом вычислительных ядер, способных работать в параллельном режиме и использовать для решаемой задачи огромные ресурсы всей системы. Суперкомпьютеры позволяют применять максимально детализированные модели для решаемых задач, использовать пространственную сетку большой размерности и получать результат за небольшое расчетное время.

Задача на собственное значение (нахождение $K_{\text{эфф}}$) является одной из основных задач нейтронной физики активной зоны реактора. Повышенные требования к точности решения этой задачи определяются критериями безопасности в работе ЯЭУ (ядерно-энергетических установок).