УДК 519.6

ОПЫТ РАЗРАБОТКИ КОМПЛЕКСА ЛАБОРАТОРНЫХ РАБОТ ПО МЕТОДАМ ВЫЧИСЛЕНИЙ НА ПЛАТФОРМЕ .NET

О.Г. Савихин

Нижегородский госуниверситет им. Н.И. Лобачевского

savihin@list.ru

Поступила в редакцию 18.11.2014

Рассматривается задача разработки единой среды пользователя, которая позволяла бы проводить параллельный расчет нескольких вычислительных экспериментов и обеспечивала единообразный подход к вводу исходных данных и отображению результатов.

Ключевые слова: образовательные программные комплексы, лабораторные работы, методы вычислений, среда пользователя, платформа .NET, язык программирования C#.

Введение

Для обучения студентов предметам, связанным с компьютерными вычислениями, кроме коммерческих пакетов прикладных программ, таких как MATLAB, с успехом используются отечественные специализированные программные комплексы, например ПараЛаб [1]. Они позволяют приобрести практические навыки и предоставляют широкий спектр возможностей для проведения вычислительных экспериментов.

На кафедре теоретической механики механико-математического факультета ННГУ для студентов, обучающихся по специальности «Прикладная математика», в составе образовательного комплекса [2] разработаны лабораторные работы по курсу «Методы вычислений». По каждой теме курса были созданы отдельные приложения (например, [3]). Данные приложения позволяют наглядно представить результаты решения при довольно комфортном интерфейсе пользователя, но они имеют также ряд недостатков. Основным из них является отсутствие интеграции приложений друг с другом, необходимых, например, для сравнения решений, полученных разными способами. Также невозможно сравнить решения разных систем, решаемых одним способом. Кроме того, программы имеют различный интерфейс, поэтому у пользователя могут возникнуть трудности при переходе от одной программы к другой.

В связи с этим возникла задача разработки единой среды пользователя, которая позволяла бы проводить параллельный расчет нескольких вычислительных экспериментов и обеспечивала единообразный подход к вводу исходных данных и отображению результатов. Пользователь среды, обладающей такими свойствами, получает возможность легко сравнивать промежуточные результаты нескольких вычислительных потоков в одном масштабе времени. Более того, среда должна предоставлять точный инструмент для сравнения скорости решения в зависимости от выбора различных методов либо от различных параметров одного метода. В качестве одного из критериев эффективности метода может служить затраченное процессорное время. Создаваемая пользовательская среда должна обладать архитектурой, позволяющей добавлять новые вычислительные методы и графические компоненты для вывода результатов без перекомпиляции самого приложения.

Кроме возможности проведения вычислительных экспериментов среда должна оказать помощь студентам в освоении элементов современных технологий программирования. Помимо создания собственной программной реализации вычислительного метода в виде внешней задачи и сравнения результатов решения в среде с эталонным методом, студент должен реализовать метод в виде динамически подключаемой библиотеки, а интерфейс к нему – в виде специального элемента управления.

Разработка единой среды пользователя «Лабораторный практикум по методам вычислений» была проведена студентами кафедры в ходе выполнения многочисленных курсовых и дипломных работ. В них решались задачи, связанные с определенной технологией программирования. Полученные результаты использовались на спецкурсах студентами 4-го и 5-го годов обучения при создании конкретных вычислительных методов среды.

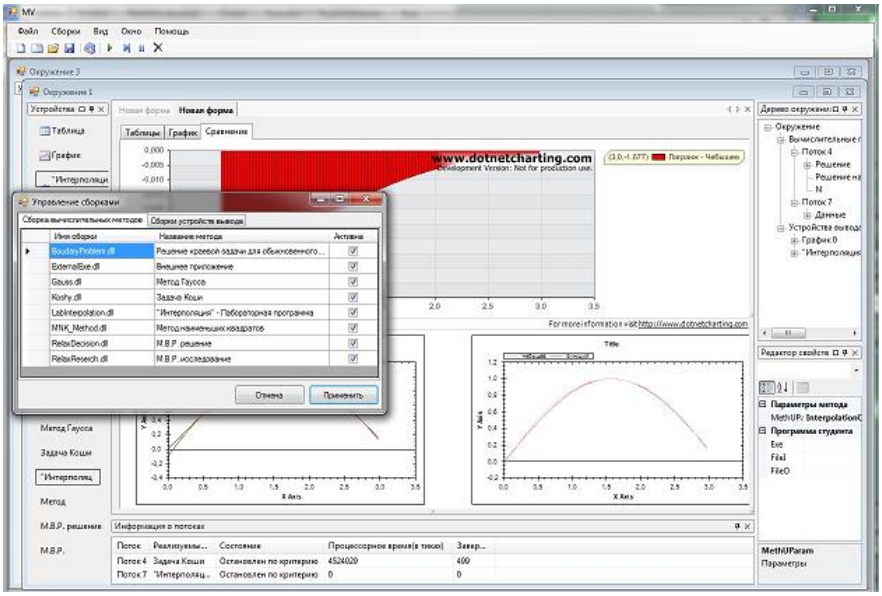

Рис. Интерфейс среды пользователя

Архитектура Microsoft .NET позволяет разрабатывать приложения на нескольких языках программирования и обеспечивает однотипность проведения многих операций на всех этих языках. В качестве языка программирования ядра среды «Лабораторный практикум по методам вычислений» был выбран C# [4]. В силу того, что он специально был создан для новой платформы, многие операции могут быть проведены более компактно, при этом C# обладает всеми механизмами объектно-ориентированного программирования. Программная оболочка использует так называемые сборки – двоичные файлы DLL (Dynamic Link Library). Эти файлы содержат метаданные, типы и дополнительные ресурсы. Платформа .NET даёт возможность писать библиотеки на любом .NET-совместимом языке. Все вычислительные методы среды реализованы в виде динамических библиотек. С каждым методом можно связать один или несколько визуальных элементов управления, на которых будут отображаться результаты решения. Элемент управления может быть стандартным или пользовательским, находящимся в динамической библиотеке. Для того чтобы методы и пользовательские элементы управления могли взаимодействовать с программной оболочкой, они реализуются в виде классов, наследующих интерфейс абстрактных базовых классов. Базовые классы, в свою очередь, определены вне среды во внешней сборке. Используемая структура построения среды не требует изменения в коде исходных файлов и перекомпиляции при подключении новых библиотек методов или пользовательских элементов управления. Это делает среду легко расширяемой и настраиваемой.

Интерфейс среды пользователя

Интерфейс среды пользователя «Лабораторный практикум по методам вычислений» содержит меню, панель с кнопками, дублирующие команды пунктов меню, рабочую область, содержащую графические компоненты для вывода данных, а также пять плавающих панелей, по функциональности аналогичных тем, что используются в VisualStudio.

По умолчанию слева вверху располагается панель со списком доступных элементов управления, используемых для отображения результатов расчетов, которые для простоты назовём устройствами вывода. Устройства вывода могут быть представлены компонентом из панели ToolBox среды VisualStudio, графиком, таблицей, составным устройством вывода или пользовательским элементом управления. Слева внизу находится панель со списком доступных методов. Сборки устройств вывода и методов вычисления при запуске основного приложения ищутся в виде файлов с расширением .dll в папках OutDevices и Methods и их названия отображаются в соответствующих списках на панелях. С целью облегчения управления сборками вычислительных методов и выходных устройств в базовом приложении был создан мастер, позволяющий подключать не все, а лишь только необходимые сборки. Справа вверху находится дерево вычислительных пото-

ков методов. Оно отображает связи потоков с выходными данными методов и устройствами вывода, на которых выводятся эти данные. Панель внизу главного окна содержит текущую информацию о потоках: название реализуемого метода, состояние (остановлен по критерию, пауза, не запущен), процессорное время, количество итераций. Графическая среда пользователя имеет собственный редактор свойств, который обеспечивает взаимодействие с объектами приложения. Все объекты могут легко и наглядно создаваться, редактироваться, храниться и удаляться. Кроме стандартного редактора свойств, реализованного на основе компонента PropertyGrid, существуют специализированные редакторы, которые, например, используются для заполнения матриц.

Порядок работы

Для добавления нового устройства вывода на форму необходимо выбрать нужное устройство из списка левым щелчком мыши, а затем щелкнуть на свободное место на форме рабочей области. С помощью рамки можно перемещать устройство вывода по форме, менять размеры, а двойным щелчком расширить его до размеров формы и обратно. Щелчок на вычислительном методе добавляет вычислительный поток в панель внизу приложения, где отображается текущая информация о потоках, а также в панель дерева потоков методов. Чтобы отобразить будущие результаты вычислений, нужно выбрать из дерева потоков необходимые выходные данные и перетащить мышью выбранный элемент на одно из устройств вывода, уже находящееся на форме рабочей области. Выбрав какой-либо поток из списка или дерева потоков, можно задать его свойства в панели свойств внизу справа, а затем запустить поток кнопкой в верхней панели. Потоки можно приостанавливать, удалять, выполнять пошагово, причем эти действия можно выполнить сразу для нескольких потоков – выбрать группу потоков можно с помощью мыши и зажатой клавиши Ctrl. При работе в приложении с большим количеством данных и устройств вывода была реализована возможность сохранения для последующего использования конфигурации рабочей области в файл и загрузки из него.

Задачи, возникающие в процессе разработки, и используемые технологии программирования

1. Подключение сборок устройств вывода и методов вычисления. При разработке приложения важную роль сыграло использование технологии .NET Reflection. При помощи данной технологии появилась возможность создавать объекты классов, определенных в подключаемых библиотеках (сборках), во время выполнения приложения (динамически).

2. Создание плавающих панелей. Гибкий интерфейс приложения достигается благодаря возможности перемещать, группировать, скрывать в удобной форме отдельные элементы управления. При создании плавающих панелей использовалась библиотека MagicLibrary. Использование библиотеки MagicLibrary связано с тем, что стандартные компоненты среды разработки Visual Studio 2005 не обладают возможностями для создания плавающих форм и панелей.

3. Сохранение и загрузка рабочей области. Сохранение и загрузка рабочей области производится при помощи механизма сериализации и десериализации экземпляров класса в виде XML документов.

4. Соединение выходных данных вычислительного метода с устройством вывода. При решении этой задачи использовалась технология Drag-and-drop. Для того чтобы устройство вывода могло поддерживать эту технологию, оно должно реализовать заданный интерфейс, определенный в базовом классе.

5. Запуск из среды внешнего приложения и отображение его результатов работы. Задача решалась с помощью запуска в среде нового процесса и обмена данными при помощи файла. Путь к внешнему приложению и к файлу с данными, а также формат самих данных задается в редакторе свойств.

6. Реализация многопоточности. В программе выполняется один главный поток, отвечающий за создание дочерних окон, обработку сообщений и за интерактивное взаимодействие с пользователем. Кроме того, приложение создает несколько вычислительных потоков. Обмен данными происходит через буфер, реализованный в виде очереди. Синхронизация потоков осуществляется по схеме классической задачи «потребитель-производитель» с использованием средств языка программирования C#.

Заключение

В работе рассмотрена задача разработки единой среды пользователя, которая позволяет проводить параллельный расчет нескольких потоков и обеспечивает единообразный подход к вводу исходных данных и отображению результатов. Среда предоставляет широкий спектр возможностей для проведения вычислительных экспериментов. Она позволяет не только приобрести практические навыки, но и оказывает по-

мощь студентам в освоении элементов современных технологий программирования.

Список литературы

1. Лабутина А.А., Гергель В.П. ПараЛаб – среда для проведения вычислительных экспериментов для изучения и исследования параллельных методов решения сложных вычислительных задач// Вестник Нижегородского университета им. Н.И. Лобачевского. № 3 (2). Н. Новгород: Изд-во ННГУ им. Н.И. Лобачевского, 2011. С. 239–247.

2. http://www.itlab.unn.ru/ (дата обращения 05.02.2014).

3. http://www.itlab.unn.ru/uploads/num/Gauss.pdf (дата обращения 05.02.2014).

4. Троелсен Э. C# и платформа .NET. Библиотека программиста. СПб.: Питер, 2004.

THE EXPERIENCE OF DEVELOPMENT OF A SET OF LABORATORY WORKS ON METHODS OF CALCULATIONS ON THE .NET PLATFORM

O.G. Savikhin

We consider the problem of establishing a common user environment, which would allow parallel calculation of several computational experiments and provide a unified approach to data input and display of results.

Keywords: educational software systems, laboratory work, methods of calculations, user environment, .NET platform, C# programming language.

References

1. Labutina A.A., Gergel' V.P. ParaLab – sreda dlia provedeniia vychislitel'nykh eksperimentov dlia izucheniia i issledovaniia parallel'nykh metodov resheniia slozhnykh vychislitel'nykh zadach// Vestnik Nizhegorodskogo universiteta im. N.I. Lobachevskogo. № 3 (2). N. Novgorod: Izd-vo NNGU im. N.I. Lobachevskogo, 2011. S. 239–247.

2. http://www.itlab.unn.ru/ (data obrashcheniia 05.02.2014).

3. http://www.itlab.unn.ru/uploads/num/Gauss.pdf (data obrashcheniia 05.02.2014).

4. Troelsen E. S# i platforma .NET. Biblioteka programmista. SPb.: Piter, 2004.