УДК 004.4'23

В.А. Карапетьян, канд. техн. наук, доцент
Севастопольский национальный технический университет,
ул. Университетская 33, г. Севастополь, Украина, 99053
E-mail: vak@stel.sebastopol.ua

ОРГАНИЗАЦИЯ ИНТЕРФЕЙСА МЕЖДУ DELPHI-ПРИЛОЖЕНИЕМ И СРЕДСТВАМИ СИСТЕМЫ MATLAB

В статье рассматриваются способы реализации программных интерфейсов между Delphi-приложениями и средствами системы Matlab на основе использования математических DLL-библиотек Matlab, встроенного механизма Matlab Engine, стандартных средств Windows для поддержки технологии Автоматизации.

***Ключевые слова**: приложение, Delphi, Matlab, межпрограммный интерфейс, автоматизация.*

Разработка пользовательских приложений для решения задач анализа и синтеза систем автоматического управления, статистической и спектральной обработки информации, моделирования поведения динамических систем и т. п. практически всегда связана с реализацией сложных математических алгоритмов. Это векторно-матричный и полиномиальный анализ, решение систем обыкновенных дифференциальных уравнений, спектральный и корреляционный анализ. Программная реализация таких алгоритмов требует наличия специализированных библиотек процедур и функций с надёжной и эффективной численной математикой. В стандартных системах программирования этих средств нет.

В настоящее время существует несколько профессиональных систем компьютерной математики, таких как MathCAD, Matlab и др., возможности которых обеспечивают эффективное решение указанных выше задач. Однако использование этих систем требует от пользователя, помимо хорошего понимания собственно предметной области, весьма глубоких знаний, как самой системы, так и наличия необходимого опыта работы в ней. Обычному пользователю, в этом смысле, весьма непросто реализовывать сложные вычислительные проекты. Следует также заметить, что в настоящее время профессиональные приложения прикладного характера должны иметь развитый графический интерфейс со всеми атрибутами дружественности – контролем ввода данных, использованием соответствующей терминологии, встроенной системой помощи и т. п. Поэтому возникает актуальная задача разработки приложений со сложной математической функциональностью и с предметно-ориентированным дружественным графическим интерфейсом.

Для разработки приложений с профессиональным графическим интерфейсом имеются вполне адекватные программные инструментальные средства – системы визуального программирования, например, Delphi. Однако эти системы программирования располагают весьма ограниченным набором процедур математического характера. Данный недостаток систем программирования можно нивелировать, используя возможности специализированных математических пакетов.

Разработчиками системы Matlab предлагаются несколько вариантов использования средств Matlab в автономных приложениях или, наоборот, расширения возможностей Matlab объектами, созданными в системах программирования. Реализованы и документированы средства организации связи с Matlab для приложений, написанных на языках Fortran, C/C++, Java, C#. Имеется и соответствующая литература по этому вопросу [1, 2]. Однако в ней в основном пересказывается содержание фирменной документации, организация же взаимодействия Delphi и Matlab в указанном выше контексте не описана.

Цель статьи: рассмотреть несколько способов организации использования Delphi-приложениями средств системы Matlab: обращение к DLL-библиотекам с математическими функциями ядра Matlab; применение механизма Matlab Engine; реализация технологии Автоматизации стандартными средствами операционной системы Windows.

В общем виде приведенные выше способы организации межпрограммных интерфейсов, описать довольно сложно. Проще проиллюстрировать предлагаемые способы отдельными простыми примерами Delphi-приложений.

Как известно, Matlab поддерживает работу с несколькими основными типами данных — числовыми массивами, строками, ячейками, структурами [3]. Типом данных по умолчанию является двумерный числовой массив (вещественный или комплексный массив с элементами типа Double). Практически все математические функции Matlab работают с этим типом данных. Поэтому в тестовых Delphi-примерах рассмотрена работа исключительно с данными типа Double. Использование других типов данных не вызывает дополнительных трудностей.

Назначение тестового Delphi-приложения состоит в выполнении следующих операций:

а) ввод пользователем исходных числовых данных в контексте языка Object Pascal;

б) преобразование Pascal-данных к числовому типу Matlab-данных;

в) передача преобразованных данных тому или иному средству Matlab для выполнения выбранной математической операции над этими данными;

г) возврат результата математической операции в Delphi-приложение;

д) обратное преобразование Matlab-данных в соответствующие им типу Pascal-данные;

е) визуализация средствами Delphi результатов математических расчётов в числовой или графической форме.

Выполнение первого и последнего пунктов не вызывает особых вопросов. Организация механизма исполнения остальных пунктов представляет собой собственно интерфейс между Delphi-приложением и системой Matlab. Для поддержки этого межпрограммного интерфейса в Delphi-приложении должны быть сформированы адекватные языковые объекты — типы данных, переменные, функции и процедуры. Все данные объекты удобно расположить в едином модуле (unit) и по мере необходимости наполнять его новыми программными объектами. Назовём этот модуль модулем сопряжения средств Matlab и системы программирования Delphi.

Для подготовки числовых массивов для Matlab в этом модуле необходимо объявить следующие типы данных:

```
type
  pDouble = ^Double;
  mxComplexty = (mxREAL, mxCOMPLEX);
  pmxArray = ^mxArray;
  mxArray  = record   end;
```

Здесь pDouble — типизированный указатель на вещественный 8-байтовый тип Double; mxComplexty = (mxREAL, mxCOMPLEX) — перечислимый тип для определения вещественного или комплексного характера числового массива; mxArray — тип-запись для обозначения особой (пустой) структуры данных, описывающий числовой массив; pmxArray — указатель на запись mxArray.

Основную нагрузку здесь несёт тип-запись mxArray. В нашем случае этот тип достаточно обозначить в виде пустой записи! Переменные данного типа создаются специальными функциями из DLL-библиотек Matlab [1]. При создании такой переменной, поля записи автоматически заполняются необходимой информацией и обращения к ним также обеспечивают специальные функции DLL-библиотек Matlab.

Все версии системы Matlab вплоть до последней (Matlab 2010b) имеют в своём составе особую DLL-библиотеку libmx.dll, которая содержит С-функции, предназначенные для манипуляций с Matlab-данными — создание и удаление массивов данных, доступ к элементам массивов и к их атрибутам. Назначение и краткие описания основных функций этой библиотеки приведены в литературе [1, 2]. Для обращения к этим функциям из Delphi-приложения в модуле сопряжения необходимо описать их заголовки в стиле языка Object Pascal. Например, функция, выделяющая необходимый объем памяти под двумерный массив комплексных чисел, будет иметь в модуле сопряжения следующий заголовок:

```
function mxCreateDoubleMatrix(m,n : Integer; mxData : mxComplexty):pmxArray;
                                                    cdecl;  external 'libmx.dll';
```

В модуль сопряжения следует добавить ещё несколько функций/процедур, необходимых для манипуляций с числовыми Matlab-массивами и имеющих следующие заголовки:

```
procedure  mxSetName(arr_ptr : pmxArray; const name : String);  cdecl;  external 'libmx.dll';
function  mxGetPr(arr_ptr : pmxArray):pDouble;  cdecl;  external 'libmx.dll';
function  mxGetPi(arr_ptr : pmxArray):pDouble;  cdecl;  external 'libmx.dll';
procedure mxDestroyArray(arr_ptr : pmxArray);  cdecl;  external 'libmx.dll';
procedure mxFree(var ram : Pointer);  cdecl;  external 'libmx.dll';
function mxTranspose(arr_ptr : pmxArray):pmxArray;  cdecl;  external 'libmx.dll';
```

Назначение этих функций определяется их названиями — присвоение имени массиву данных, получение указателей на вещественную и мнимую части массива данных, уничтожение массива и освобождение памяти, транспонирование массива.

Описания всех заголовков должны содержать директиву cdecl (или stdcall) для согласования различных способов обработки параметров функций, присущих языкам программирования Pascal и C.

Опишем порядок действий при выполнении пунктов б) и д) — преобразования Pascal-данных в Matlab-данные и обратно — Matlab-данных в Pascal-данные.

Рассмотрим выполнение прямого преобразования Pascal → Matlab. Допустим, что имеется двумерный $n \times m$ комплекснозначный массив Z, вещественная часть которого содержится в массиве X, а

мнимая — в массиве Y, т.е. $Z = X + iY$. Для удобства дальнейших действий переформатируем двумерные массивы X и Y по строкам в одномерные массивы. Затем выполним следующие действия:

— выделяем память под $n \times m$ комплекснозначный массив и именуем его как Z

```
arrZ_ptr := mxCreateDoubleMatrix(n, m, mxCOMPLEX);
mxSetName(arrZ_ptr, 'Z');
```

— копируем побайтно элементы вещественной (X) и мнимой (Y) частей массива в отведённую часть памяти для массива Z

```
Move(X[0], mxGetPr(arrZ_ptr)^, n*m*sizeof(Double));
Move(Y[0], mxGetPi(arrZ_ptr)^, n*m*sizeof(Double));
```

— транспонируем двумерный массив Z, так как операции с ним будут проводить функции Matlab, которые работают с двумерными массивами в стиле языка C,

```
arrZ_ptr := mxTranspose(arrZ_ptr);
```

Теперь комплекснозначный массив Z (два вещественных Pascal-массива X и Y) конвертирован в структуру данных, корректно воспринимаемую системой Matlab как обычный массив комплексных чисел. Указатель на эту структуру содержится в переменной arrZ_ptr.

Обратное преобразование Matlab → Pascal выполняется следующим образом Допустим, что имеется одномерный $1 \times n$ вещественный Matlab-массив, на который указывает arrM_ptr. В Delhi-приложении для его приёма приготовлен обычный одномерный $1 \times n$ вещественный массив Mas, например, определённый как var Mas : array of Double. Операция обратного конвертирования осуществляется, например, следующей последовательностью Pascal-инструкций:

```
SetLength(Mas,n);
Move(mxGetPr(arrM_ptr)^, Mas [0], n*sizeof(Double));
```

Подготовленные в формате mxArray числовые массивы могут быть переданы тем или иным средствам Matlab для выполнения математических операций над ними. Рассмотрим кратко некоторые способы обращения к этим средствам из приложений Delphi.

1. Использование математической библиотеки C/C++

Вплоть до седьмой версии математические процедуры общего назначения системы Matlab были оформлены в виде математической библиотеки C/C++. Начиная с версии Matlab 7.0, математическая библиотека C/C++ была заменена на среду выполнения компонентов Matlab MCR (Matlab Component Runtime).

Математическая библиотека Matlab 6.x включает в себя более 400 математических процедур и реализована в виде набора DLL-библиотек из 48-ти файлов. Для свободного распространения и использования все файлы библиотеки упакованы в архив mglinstaller.exe. Процедуры математической библиотеки написаны на языках C и C++. Поэтому для использования их в приложениях Delphi необходимо корректным образом прописать заголовки необходимых процедур в модуле сопряжения. Имена всех процедур в библиотеке начинаются с префикса mlf. Например, процедура вычисления собственных векторов и собственных чисел матрицы будет иметь в Delphi-модуле следующий заголовок:

```
function mlfEig(D,A,B:pmxArray):pmxArray; cdecl; external 'libmatlb.dll';
```

Здесь указано, что тело функции mlfEig расположено во внешней (external) DLL-библиотеке libmatlb.dll.

Практически весь набор функций математической библиотеки Matlab описан в контексте языка C в фирменной документации [4]. Корректное представление заголовков C-функций в терминах языка Pascal, с учетом описанных выше типов данных, позволяет использовать в приложениях Delphi всю мощь ядра математики Matlab.

Опыт применения данного механизма организации связи Delphi-приложений и ядра математики Matlab доказал свою эффективность во множестве конкретных приложений. Однако этот механизм не позволяет использовать математические средства Matlab, расположенные в его специализированных пакетах — Toolboxes. Довольно часто именно там и находятся наиболее интересные и ценные математические процедуры, необходимые для решения конкретной специальной задачи. Следующий механизм позволяет ликвидировать этот недостаток.

2. Использование технологии Matlab Engine

Данная технология, предлагаемая самими разработчиками Matlab, позволяет получить программный доступ не только к математическому ядру Matlab, но и практически ко всем его специализированным пакетам. Правда за это приходится платить большую цену – предполагается

полный доступ к системе Matlab. Для индивидуального пользователя это означает наличие установленного Matlab в его компьютерной системе.

Суть данной технологии заключается в том, что приложение, когда в этом возникает необходимость, инициирует (загружает) Matlab и управляет его работой — передаёт в рабочую область данные, пересылает в командное окно необходимые команды, возвращает в приложение данные из рабочей области, деинициирует (выгружает) Matlab.

Технология Matlab Engine представляет собой одно из расширений общей технологии COM, а именно — Автоматизацию. В данном случае Matlab выступает в роли сервера автоматизации, а Delphi-приложение — в роли диспетчера (контроллера) автоматизации. Интерфейсы COM-объектов обёрнуты в несколько C-функций, которые организованы в DLL-библиотеку libeng.dll.

Для доступа приложения Delphi к Matlab Engine необходимо в Pascal-модуль сопряжения добавить описания ещё двух специфических типов данных:

```
type
  Engine = record   end;
  pEngine =^Engine;
```

и, как минимум, пяти функций из библиотеки libeng.dll:

```
function engOpen(par : pointer):pEngine;      cdecl;   external 'libeng.dll';
function engClose(par : pEngine):pEngine;     cdecl;   external 'libeng.dll';
function engGetVariable(ep : pEngine; const var_name : PChar):pmxArray;
                                                       cdecl;   external 'libeng.dll';
function engPutVariable(ep : pEngine; st : PChar; data : pmxArray):pEngine;
                                                       cdecl;   external 'libeng.dll';
function engEvalString(ep : pEngine; const parst : PChar):pEngine;
                                                       cdecl;   external 'libeng.dll';
```

Указатель на пустую строку pEngine идентифицирует загруженную приложением копию Matlab. Первые две функции применяются для загрузки и выгрузки Matlab; следующие две для обмена данными между приложением и рабочей областью Matlab; последняя функция служит для передачи в командное окно Matlab командной строки на выполнение операции.

Обращение из Delphi-приложения к Matlab (передача строки-команды) имитирует работу пользователя с системой Matlab в режиме суперкалькулятора (в режиме командной строки). Передаваемые в Matlab и получаемые Delphi-приложением из Matlab'a данные упакованы в структуры типа mxArray. Pascal-модуль сопряжения, обеспечивающий механизм Matlab Engine, должен содержать исключительно функции из DLL-библиотек libmx.dll и libeng.dll. Эти библиотеки входят в состав всех версий Matlab.

Технология Matlab Engine является фирменной (MathWorks) оболочкой механизма Automation, поддерживаемого операционной системой Windows. Естественно, возникает желание непосредственно реализовать этот системный механизм для организации межпрограммного интерфейса между Delphi-приложением — клиентом и Matlab — сервером. Система программирования Delphi имеет в своём составе средства организации такого механизма. Это факт и позволяет реализовать в "чистом виде" технологию Автоматизации.

3. Использование системы Matlab как сервера Автоматизации

Для работы с Matlab как с сервером Автоматизации необходимо знать свойства и методы (интерфейсы), которые предоставляет сервер Автоматизации. Эта информация доступна из фирменной документации [4].

Обмен данными между Delphi-приложением и Matlab со стороны Delphi-приложения (диспетчера Автоматизации) осуществляется с помощью переменных вариантного типа (тип OLEVariant). Matlab, как сервер Автоматизации, предоставляет Delphi-приложению свои функции PutFullMatrix и GetFullMatrix для обмена числовыми массивами, а также функцию Execute для передачи команды на выполнение операции сервером Автоматизации. Функции CreateOleObject и Quit соответственно открывают и закрывают сервер (Matlab).

В качестве примера данного способа организации связи Delphi-приложения с Matlab приведём фрагменты Delphi-кода для решения задачи вычисления ранга вещественной $n \times m$-матрицы.

Определим необходимые переменные:

```
var
  Mtlb : Variant;
  A : array of array of Double;
  A_re, A_im, R_re, R_im  :  OLEVariant;
```

Переменные предназначены для идентификации сервера Автоматизации (Mtlb), хранения исходной двумерной матрицы (A), хранения вещественной (A_re) и мнимой (A_im) частей матрицы A, хранения вещественной (R_re) и мнимой (R_im) частей значения ранга матрицы A соответственно.

Исполняемые операторы фрагмента кода:

```
{1}   SetLength(A,n,m);
{2}   for i:=1 to n do
{3}     for j:=1 to m do  A[i-1,j-1]:=random;
{4}   A_re:=VarArrayCreate([0,n-1,0,m-1],varDouble);
{5}   A_im:=VarArrayCreate([0,N-1,0,M-1],varDouble);
{6}   for i:=0 to n-1 do
{7}     for j:=0 to m-1 do
{8}       begin A_re[i,j]:=A[i,j]; A_im[i,j]:=0 end;
{9}   Mtlb:=CreateOleObject('Matlab.Application');
{10}  Mtlb.PutFullMatrix('A','base',VarArrayRef(A_re),VarArrayRef(A_im));
{11}  Mtlb.Execute('r = rank(A);');
{12}  R_re:=VarArrayCreate([1,1],varDouble);
{13}  R_im:=VarArrayCreate([1,1],varDouble);
{14}  Mtlb.GetFullMatrix('r','base',VarArrayRef(R_re),VarArrayRef(R_im));
{15}  Mtlb.Quit;
```

Здесь в строках 1÷3 выделяется память под массив A и генерируются значения элементов массива A. Инструкции в строках 4÷8 формируют и заполняют два двумерных вещественных вариантных массива A_re и A_im под вещественную и мнимую части матрицы A. В строке 9 диспетчер Автоматизации (Delphi-приложение) инициирует сервер Автоматизации (Matlab). Операторы в строках 10 и 11 предназначены соответственно для передачи массива A в рабочую область и выполнения команды вычисления ранга матрицы A. Инструкции 12-ой и 13-ой строк готовят переменную вариантного типа для приёма значения ранга матрицы из рабочей области Matlab. В 14-ой строке происходит передача значения переменной Matlab r (ранга матрицы) в вариантную переменную (R_re, R_im) приложения Delphi. Инструкция в строке 15 закрывает сервер Автоматизации.

Рассмотренные выше способы организации межпрограммного интерфейса Delphi-приложений и средств системы Matlab были неоднократно реализованы в практической работе и показали свою надёжность и эффективность. Тестировались предлагаемые способы в диапазонах: по Delphi-измерению — от Delphi® 5 до Embarcadero® Delphi® 2010; по Matlab-измерению — от Matlab® 6.0 до Matlab® R2010b.

Дальнейшее направление работы — использование последних версий пакета Matlab Compiler для создания пользовательских DLL-библиотек из m-функций и подключения их к приложениям Delphi.

Библиографический список используемой литературы

1. Подкур М.Л. Программирование в среде Borland C++ Builder с математическими библиотеками Matlab C/C++ / М.Л. Подкур, П.Н. Подкур, Н.К. Смоленцев. — М.: ДМК Пресс, 2006. — 496 с.

2. Смоленцев Н.К. MATLAB и программирование на Visual C#, Borland JBuilder, VBA: Учебный курс (+CD) / Н.К. Смоленцев. — М.: ДМК Пресс; СПб.: Питер, 2009. — 464 с.

3. MATLAB 7. Programming Fundamentals. — MathWorks, Inc., 2010. — 684 p.

4. MATLAB 7. External Interfaces. — MathWorks, Inc., 2010. — 776 p.

Поступила в редакцию 2.12.2010 г.

Карапетьян В.А. Організація інтерфейсу між Delphi-програмами та засобами системи Matlab

У статті розглядаються способи реалізації програмних інтерфейсів між Delphi-програмами та засобами системи Matlab на основі використання математичних DLL-бібліотек Matlab, вбудованого механізму Matlab Engine, стандартних засобів Windows для підтримки технології автоматизації.

Ключові слова: програма, Delphi, Matlab, міжпрограмний інтерфейс, автоматизація.

Karapetyan V.A. The interface organization between a Delphi-application and the MATLAB system

Means of realization of program interfaces between Delphi-applications and the system of Matlab on the basis of using the mathematical Matlab DLL-libraries, the built-in mechanism Matlab Engine, standard means of Windows for supporting the technology of Automation are considered.

Keywords: application, Delphi, Matlab, program interface, Automation.