УДК 664.71

А.В. Романов, К.И. Коломиец

ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ КОМПОНЕНТЫ ДЛЯ РАЗРАБОТКИ ПРИКЛАДНЫХ РАСЧЕТНЫХ ПРОГРАММ

Вопросы разработки современных прикладных расчетных программ, ориентированных на автоматизированное проектирование технических систем (ТС) и реализующих стандартный Windows-интерфейс, в научно-технической литературе практически не рассматриваются. Можно отметить следующую закономерность. С одной стороны, существуют литература (например, [1]), в которой довольно подробно разбираются конкретные алгоритмы автоматизированного расчета, но приводимые примеры отражают лишь математические особенности расчетов и ориентированы на языки программирования типа Бейсик, Паскаль, Фортран, которые, хотя и являются основой современных объектно-ориентированных языков программирования, но едва ли могут с ними отождествляться. С другой стороны, значительно большее количество книг, например [2, 3, 4, 5], приводят методики и примеры объектно-ориентированного программирования вообще, без ориентации на конкретную предметную область.

Задачи разработки прикладных расчетных программ, в том числе и для автоматизированного проектирования, с точки зрения программирования имеют свои особенности, которые нуждаются в исследовании и анализе. В частности, это достоинства и недостатки представления проектируемой технической системы в виде специального объекта или класса, когда многие математические расчеты будут "спрятаны" в методы доступа к определенным параметрам ТС (полям класса); вопросы обязательного информационного обеспечения прикладных программ; достоинства и недостатки различных способов организации интерфейса расчета и многое другое.

Типичными задачами прикладного ПО являются ввод и визуализация (вывод на экран) логических, строковых, числовых данных (последние для ТС наиболее распространены), причем при выводе данных желательно снабжать их соответствующими комментариями, которые как правило имеют вид "сложного" текста. Под термином "сложный" подразумевается присутствие в тексте различных шрифтов, стилей, надстрочных и подстрочных символов и т.д. (Сравните, например, два описания – "Момент инерции J1, кг*м^2" и "Момент инерции J_1, кг*$м^2$".) Отсутствие соответствующих компонентов или сложность адаптации стандартных компонентов является одной из проблем разработки прикладного программного обеспечения в среде Delphi.

Авторами были разработаны два компонента.

Компонент для ввода численных параметров. В основу положен стандартный компонент поля ввода (класс TEdit). Он был дополнен свойствами (property) для хранения собственно значение вводимого параметра; для определения верхней и нижней границ допустимого диапазона ввода и флагами для разрешения ввода отрицательных и целых чисел; а также процедурами определения выхода введенного значения за заданные границы. Также был переопределен метод класса KeyPress, реализующий прием компонентом кода нажатой клавиши клавиатуры. Новый компонент TNEdit позволяет пользователю вводить только:

- численные клавиши, "+", "-", "Забой", причем состояние флага ввода отрицательных чисел определяет возможность появления символа "-" в первой позиции окна ввода;
- десятичный разделитель "." или "," с корректным преобразованием через системную переменную DecimalSeparator (зависит от флага ввода только целых чисел);
- нажатие любой буквенной клавиши приводит к однократному появлению символа экспоненциальной формы числа – "E".

Если вводимое число выходит за заданный диапазон, то цвет цифр компонента меняется с черного на красный.

Таким образом, разработанный компонент TNEdit практически гарантирует корректность введенного числа при использовании его в дальнейших расчетах. Применение компонента целесообразно при использовании множества полей ввода на различных формах разрабатываемой прикладной расчетной программы. При этом результирующий код программы уменьшает незначительно, а исходный код программы на языке высокого уровня сокращается существенно.

Компонент для визуализации сложного текста. Решение можно осуществить двумя способами: внедрением в программу элемента редактора формул в виде объекта Microsoft Equation 3.0 и разработкой специального компонента.

Первый способ подразумевает использование технологии OLE. Аббревиатура OLE обозначает Objects Linked and Embedded (Присоединенные И Встроенные Объекты). Данные, разделяемые между приложениями, называются OLE объектом. Приложение, которое может содержать OLE объекты, называют OLE контейнером (OLE Container). Приложение, данные из которого можно включить в OLE контейнер в виде OLE объекта, называют OLE сервером.

Как следует из названия, OLE объекты можно либо присоединить к OLE контейнеру, либо включить в него. В первом случае данные будут храниться в файле на диске, любое приложение будет иметь доступ к этим данным и сможет вносить изменения. Во втором случае данные включаются в OLE контейнер, и только он сможет просматривать и модифицировать эти данные.

Класс TOLEContainer позволяет отображать в программе объект в его непосредственном виде (с различной степенью увеличения или уменьшения - свойство Zoom) или в виде пиктограммы.

Выбор OLE-объекта может происходить не только во время дизайна, но и во время выполнения программы. Результаты работы с этим объектом можно сохранить в виде файла и в следующий раз восстановить его оттуда, для этого TOLEContainer имеет два метода SaveToFile и LoadFromFile.

Второй способ реализуется модификацией существующего в библиотеке класса TLabel и представлен специально разработанным компонентом TFormulLabel. Это осуществляется перекрытием метода DoDrawText. Эта процедура прорисовывает текст, который находится в свойстве Caption класса TLabel. Новый метод DoDrawText написан таким образом что текст, находящийся в Caption, воспринимается им как модель, которая описывает текст как совокупность элементов.

Положение и размер каждого элемента определяется окружающими его тэгами. Внутри одного элемента может находиться произвольное количество других. Название тэга заключается в угловые скобки.

При сравнении вышеизложенных способов можно выделить их основные достоинства и недостатки. Использовать разработанный компонент необходимо и достаточно в большинстве случаев, когда появляется необходимость визуализации неоднородного теста, не содержащего специальных математических символов. При необходимости отображения сложных математических формул, необходимо использовать OLE-формулы, но для этого требуется наличие редактора формул.

Разработанные компоненты могут быть использованы не только для прикладных расчетных программ, но и, например, для разработки учебной системы контроля знаний.

Литература

1. Башарин А.В., Постников Ю.В. Примеры расчета автоматизированного электропривода на ЭВМ. Учебное пособие для вузов. – 3-е изд.–Л.: Энергоатомиздат, Ленингр. отд-ние, 1987.

2. Культин Н.Б. Программирование на Object Pascal в Delphi 5. – СПб: БВХ – Санкт-Петербург, 2000. – 464 с.

3. Тюкачев Н., Свиридов Ю. Delphi 5. Создание мультимедийных приложений. Учебный курс. – СПб.: Питер, 2001. – 400 с.

4. Архангельский А.Я. Программирование в Delphi 7. – М.: ЗАО «Издательство БИНОМ», 2003.

5. Кэнту М. Delphi 6 для профессионалов. – СПб: Питер, 2002. – 1088 с.

Воронежский государственный технический университет