УДК 004.514:004.045:004.4'2

УНИВЕРСАЛЬНАЯ ПРОГРАММНАЯ ОБОЛОЧКА TSS3 ДЛЯ ПОСТРОЕНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА ПРОГРАММНЫХ КОМПЛЕКСОВ НА ЯЗЫКЕ ФОРТРАН

Н. А. Крутько
(РФЯЦ-ВНИИЭФ)

Рассматриваются концепция и особенности реализации универсальной оболочки TSS3 для автоматизированного построения пользовательских интерфейсов программных комплексов на языке Фортран. В основу положено объектное описание структуры входных данных, трансформируемое в пользовательский интерфейс на основе сформулированных принципов соответствия и компоновки визуальных элементов редактирования. Оболочка TSS3 обладает широкими возможностями динамического управления средствами редактирования и поддерживает визуализацию данных на стадии счета.

Ключевые слова: структура данных, объектный подход, пользовательский интерфейс.

Введение

Основным языком программирования для широкого класса задач математического моделирования является Фортран. Этот язык, развиваясь на протяжении своей длительной истории, обладает в настоящее время большой выразительной силой и эффективностью при реализации вычислительных алгоритмов [1]. В то же время не вызывает сомнений трудность реализации на Фортране задач невычислительного характера, в частности, организации пользовательского интерфейса.

Проблема обеспечения вычислительных программных комплексов на языке Фортран эргономичным и дружественным пользовательским интерфейсом сохраняет постоянную актуальность со времени появления операционных систем с поддержкой графического интерфейса пользователя. До настоящего времени самым массовым типом систем с такой поддержкой остается семейство 32-разрядных систем Microsoft Windows. Наличие развитого оконного интерфейса делает эти системы весьма популярной целевой средой для подготовки входных данных и постпроцессорной обработки результатов, в то время как массовые вычисления в настоящее время перенесены на многопроцессорные вычислительные системы.

Для создания пользовательского интерфейса идеально подходят современные визуальные среды разработки на объектно-ориентированных языках, широко представленные в OS Windows и естественным образом реализующие событийную модель функционирования графического интерфейса, — например, Delphi [2]. Создаваемый при этом пользовательский интерфейс является логически автономной частью программного комплекса, тем или иным способом осуществляющей обмен с модулями — потребителями данных, написанными на языке Фортран.

В организационном аспекте имеются два альтернативных подхода к реализации пользовательского интерфейса:

- исполнение *под ключ*;
- создание универсального инструментария проектирования и построения пользовательского интерфейса, доступного для применения прикладным Фортран-программистом.

Оба подхода имеют свои преимущества и недостатки.

Исполнение пользовательского интерфейса как системы *под ключ* подразумевает максимально полный учет специфики конкретной задачи, пожеланий конечного пользователя и поэтому позволяет достичь весьма высокого комфорта при работе с программой. Кроме того,

это заведомо менее сложная задача для разработчика, поскольку она является частной. Недостаток при этом один, и он является обратной стороной преимущества, — это *жесткость* системы вследствие фиксации структуры входных данных. В этом случае изменение в спецификации отображаемых данных требует и модификации модуля пользовательского интерфейса с привлечением его разработчика, т. е. необходимо авторское сопровождение интерфейса на протяжении всего жизненного цикла программного комплекса.

Альтернативой является создание инструментальных средств, основанных на некотором общем концептуальном подходе, позволяющем программисту, реализующему вычислительную обработку данных (на языке Фортран), самостоятельно строить пользовательский интерфейс без привлечения системного программиста. Подход на основе универсальной оболочки (логически автономного компонента, реализующего пользовательский интерфейс) дает организационное преимущество, в особенности для развивающихся программных комплексов с внутренней модульной структурой, когда имеют место частые изменения в спецификации входных данных. Недостатком такого подхода является его сложность — как концептуальная, так и техническая в плане реализации. Кроме того, интерфейс, построенный на основе универсального инструментария, неизбежно связан ограничениями, накладываемыми его концепцией и реализацией.

Исходя из вышесказанного, можно очертить сферы применения обоих подходов. Для компактных счетных комплексов со стабильной спецификацией входных данных целесообразно создавать пользовательский интерфейс *под ключ*, а для развивающихся комплексов с выраженной модульной структурой оправданно построение интерфейсов с помощью универсального инструментария.

Программная оболочка TSS3 представляет собой набор инструментальных средств для построения пользовательского интерфейса задания входных данных для Фортран-программ, т. е. универсальный инструментарий в терминах вышеизложенного.

Рассмотрим подробнее основные положения концепции, реализованной в современной версии TSS3.

Концепция

Решаемая в рамках пакета TSS3 задача может быть сформулирована следующим образом: автоматизированное построение пользовательского интерфейса для задания входных данных на основе их статически описанной структуры с возможностью динамического управления представлением данных в процессе диалога с пользователем.

Структура входных данных в TSS3 описывается в рамках объектной модели. Эта модель является ядром современной концепции объектно-ориентированного программирования (ООП) и является вполне адекватной для описания данных практически любой сложности [3]. В TSS3 используется лишь *статическая* часть концепции ООП — инкапсуляция данных в системе вложенных объектов-записей, определяемых пользователем. Входные данные в TSS3 являются существенно *пассивными*, т. е. не ассоциированы с собственными, внутренними, функциями обработки информации, поэтому и не имеется аналогов методов классов и *динамической* составляющей ООП — наследования и полиморфизма.

Данные в TSS3 локализованы в определяемых пользователем именованных типах — записях (используются также термины *объект* и *класс*), которые представляют собой наборы именованных (ссылка по имени) полей элементарных встроенных и пользовательских типов, а также других записей. Кроме имени и типа, поля в записях характеризуются несколькими дополнительными атрибутами, обеспечивающими более высокий эргономический уровень графического интерфейса.

Автоматизированное построение пользовательского интерфейса основано на следующих принципах:

- скалярные объекты элементарных типов имеют текстовое (строковое) представление, редактируемое с помощью соответствующих элементов интерфейса — строкового редактора либо селекторов различных видов;
- агрегатные (составные) объекты — записи и массивы — отображаются в таблицах;
- логическая вложенность записей и массивов передается в пользовательском интерфейсе с помощью логически вложенных модальных окон.

Эти принципы в совокупности со структурой данных уже достаточны, чтобы построить статический интерфейс для редактирования входных данных по сценарию прямого доступа ко всем их элементам. Однако намного более содержательный и гибкий диалог с пользователем возможен в результате динамической связи по данным и управлению между оболочкой и модулем входных данных счетного комплекса. Последний выступает при этом в роли ведущей программы, содержащей нелинейный сценарий диалога с принятием решений и корректировкой представления данных. Результирующий диалог при этом реализует широко известную концепцию *мастера* (wizard).

В функции оболочки входит хранение данных, контроль их целостности и корректности на основе предварительно описанной структуры. Обмен данными между оболочкой и Фортран-программой происходит в обоих направлениях и технически осуществляется системными средствами межпрограммного обмена данными, скрытыми от Фортран-программиста.

Механизм для обеспечения динамического диалога заключается в обратной коррекции (предустановке) входных данных и управлении их отображением со стороны ведущей программы в зависимости от действий пользователя. Эти возможности достигаются за счет обработки событий редактирования (навигации в данных и модификации их значений) программным кодом в составе ведущей Фортран-программы. Здесь применен механизм процедур обратного вызова (callback procedures).

Кроме обеспечения редактирования входных данных, оболочка TSS3 предоставляет ведущей Фортран-программе широкий набор сервисов по организации диалога с пользователем, поскольку средства, предоставляемые в этой области стандартными библиотеками Фортрана, как правило, весьма ограничены. К числу этих сервисов относятся возможности стандартных системных диалогов (текстовые сообщения и запросы, выбор объектов файловой системы), индикация прогресса длительных операций, просмотр текстовых файлов. Оболочка TSS3 включает также два крупных функциональных блока — организацию пользовательского меню для динамического управления счетом Фортран-программы и аппарат динамического вывода графиков одномерных зависимостей.

Описание структуры данных

Описание структуры входных данных является ключевым моментом в разработке пользовательского интерфейса с помощью универсального инструментария.

Основой концепции данных в TSS3 является запись — составной именованный тип данных, определяемый пользователем как совокупность именованных логических единиц хранения данных — полей. Данные в полях записей хранятся в объектах встроенных и пользовательских типов, при этом концепция типов в целом соответствует современным представлениям ООП.

Общая схема классификации типов данных TSS3 приведена на рис. 1. Рассмотрим последовательно все эти типы данных.

К элементарным относятся типы, объекты (значения) которых являются логически неделимыми — числа, строки и логические значения.

При описании структуры данных в TSS3 используются четыре встроенных (предопределенных) элементарных типа: строка (string, длина не ограничена), логический (logical), целый (integer) и вещественный (real). Применение встроенных типов для хранения данных очевидно.

В TSS3 имеются также четыре пользовательских типа — селектор строк (stringsel), перечисление (enum), множество (set) и запись (record). Пользователь может определить и использовать при описании структуры любое количество этих типов.

Все пользовательские типы имеют два общих атрибута-строки — имя (уникально в структуре) и комментарий (описание). Имя типа является обязательным ссылочным атрибутом, комментарий — частью пользовательского интерфейса.

Среди пользовательских типов элементарными являются селектор строк, перечисление и множество. Записи представляют собой составные объекты и в совокупности описывают общую структуру входных данных.

Тип перечисления (enum) является фактически целочисленным, размером 1 байт без знака. Он предназначен для реализации одновариантного (исключительного) выбора, где каждому значению (коду) поставлена в соответствие строка описания варианта. Визуальный элемент редактирования — выпадающий список (combobox) или блок радиокнопок (radio buttons).

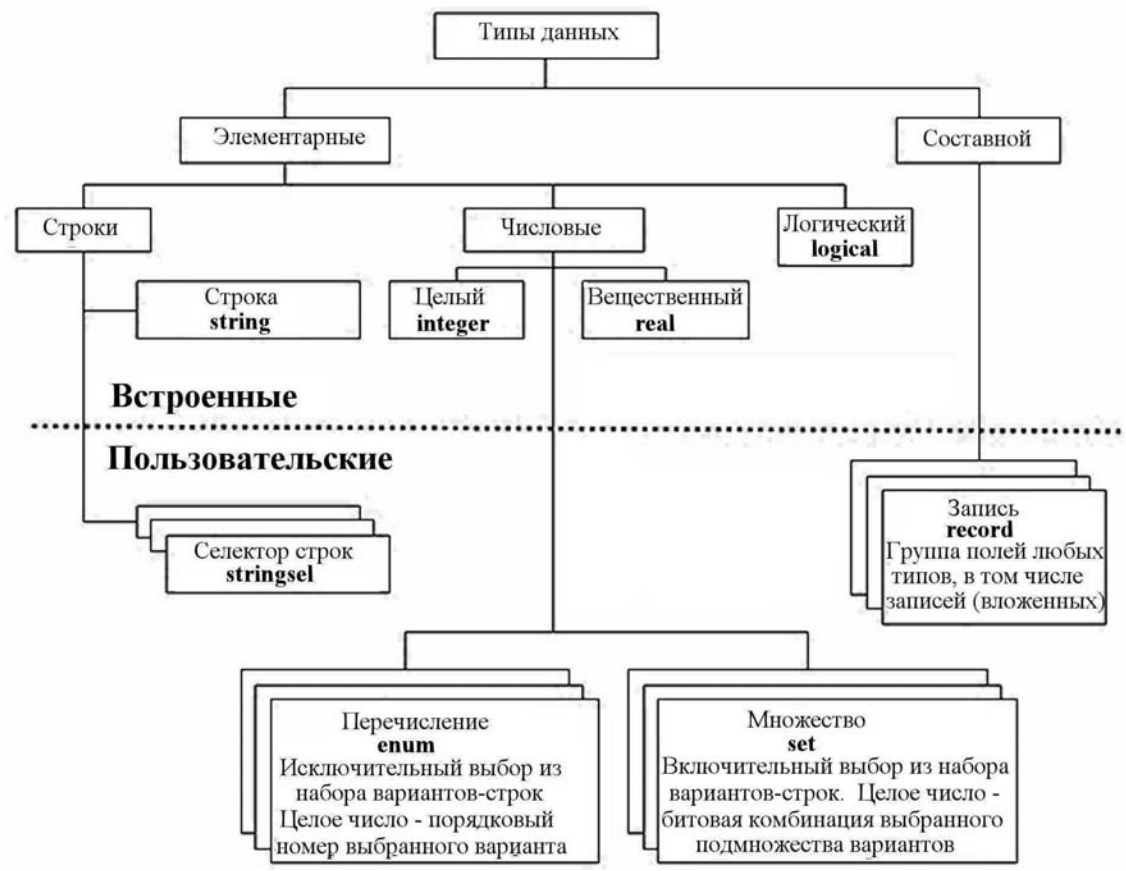

Рис. 1. Классификация типов данных TSS3

Тип селектора строк (stringsel) аналогичен перечислению, за исключением того, что объектами данных являются сами строки (а не целые числа). Число вариантов выбора не ограничено, перечень вариантов-строк может как задаваться при статическом описании структуры, так и формироваться динамически в Фортран-программе. Визуальный элемент редактирования — выпадающий список.

Тип множеств (set) является фактически 64-разрядным беззнаковым целым и предназначен для многовариантного (включительного) выбора, т. е. задания подмножеств некоторого множества элементов — строк (максимально 64, ассоциируются с битами кода). Визуальный элемент редактирования — список селекторов (checklistbox).

Запись (record) представляет собой логически единую группу объектов-полей, описывающих в совокупности определенный объект предметной области задачи. Поля записей являются именованными переменными элементарных типов либо других типов записей (рекурсивные ссылки недопустимы). Глубина вложенности записей не ограничена.

В TSS3 в памяти динамически существуют только объекты-записи, поля которых являются скалярами или массивами величин элементарных (в том числе пользовательских) типов либо объектов других типов-записей. Предусматриваются одно- и двумерные массивы любых типов, в том числе записей.

В TSS3 вся совокупность данных задачи — набор данных (dataset) — рассматривается как единственный объект-запись *корневого* типа.

Описание структуры данных в TSS3 представляет собой набор определений пользовательских типов — перечислений, селекторов строк, множеств и записей.

В рамках статически описанной структуры можно решить часть задач описания функционального поведения объектов, а именно вопросы инициализации начальных значений полей записи (в ООП — методы-конструкторы) и контроля корректности значений, присваиваемых величинам (в ООП set-методы). Поскольку описание является декларативным (отсутствуют процеду-

ры), такой контроль может быть лишь формальным (статическим) — на минимум, максимум, упорядоченность.

Определение каждого типа записи содержит одно или более полей. Общее число полей в записи не ограничено. Вся структурная информация сосредоточена в атрибутах полей записей, которые можно разделить на статические — задаваемые при описании структуры (имя поля, тип данных, размерность массива, инициализатор, диапазон значений, упорядоченность массива) и динамические — для программного управления визуализацией со стороны Фортран-программы (видимость, цвет, защита от изменений, текст подсказки).

Хранение структуры данных

Для хранения структуры данных в TSS3 применена реализация метаязыка XML (формат TSL — TSS Structure Language).

Как было сказано, структура данных формулируется как совокупность определений пользовательских типов перечислений, множеств и записей. Полное (не содержащее внешних ссылок) множество определений пользовательских типов, описывающих структуру данных одной задачи, составляет проект. Эти определения могут быть локализованы в одном или нескольких TSL-файлах, называемых модулями. В модулях группируются определения логически связанных (родственных) типов. Множество пользовательских типов в пределах модуля не обязано быть замкнутым, поэтому синтаксически поддерживается механизм импорта модулей.

Динамическая картина данных в памяти

Необходимой задачей программной оболочки TSS3 является отображение в память и динамическое поддержание в процессе работы описанной структуры данных. Как уже говорилось, все данные представляют собой единый объект-запись некоторого корневого типа.

Объекты-записи автоматически создаются в памяти по следующему сценарию. Для каждого поля записи вычисляются значения выражений размерности (имеются три варианта — скаляр, одномерный массив и двумерный массив) и выделяется соответствующий объем памяти (в зависимости от типа). Затем каждому элементу вновь созданного массива присваивается начальное значение, вычисленное по записанному в структуре выражению — инициализатору. Если типом элементов поля является (другая) запись, каждый элемент массива в текущей записи является указателем на вновь созданный объект-запись этого (другого) типа. Эти объекты, в свою очередь, создаются и инициализируются в соответствии с собственной внутренней структурой. Данный процесс осуществляется рекурсивно на всю глубину иерархии вложенных записей. В результате в начальный момент в памяти формируется система ссылочно-связанных проинициализированных объектов данных с единственным корневым объектом, соответствующая описанной структуре. Таким образом, решается задача, возлагаемая в процедурном ООП на методы-конструкторы.

Далее в процессе работы исполняются поступающие запросы по модификации данных, при этом осуществляется формальный контроль корректности данных на основе информации, описанной в структуре (минимум, максимум, сортировка). Если изменяется значение поля записи (переменной) целого типа, входящего в выражения размерностей каких-либо массивов, то эти выражения автоматически пересчитываются и размерности массивов приводятся в соответствие с новыми значениями. При этои выделяется (освобождается) память и создаются и инициализируются начальными значениями (или уничтожаются) элементы массивов. При уничтожении объекта-записи автоматически уничтожаются и все вложенные объекты-записи, рекурсивно на всю глубину.

Хранение и загрузка данных

Спецификация входных данных счетной Фортран-программы и соответственно их структура, описанная в TSL-проекте, может изменяться со временем. С другой стороны, Фортран-программа при этом эксплуатируется и непрерывно продуцируются наборы данных, каждый из которых реализует существующую на данный момент структуру. Механизм хранения данных в TSS3 обеспечивает устойчивость (понимаемую как возможность корректной интерпретации) ранее подготовленных наборов данных к изменениям их структуры, определяемой в TSL-проекте.

Ключевым подходом при этом является хранение в наборе наряду со значениями данных полной информации об их фактической структуре

(типы данных, имена пользовательских типов и полей записей, фактические размерности массивов). Таким образом, в наборе данных после его создания содержится и "слепок" TSL-проекта, по которому он был создан, за исключением визуальных атрибутов полей данных.

Наборы данных в TSS3 хранятся в двоичных файлах собственного формата TDF (TSS Data Format). Отказ от применения языка XML в данном случае продиктован соображениями скорости загрузки.

Процесс загрузки данных представляет собой по существу их интерпретацию — соотнесение структуры данных, зафиксированной в наборе при его создании, с текущей TSL-структурой. Отождествление именованных объектов структуры — пользовательских типов и полей записей — выполняется по их именам. При интерпретации данных и, в частности, при разрешении конфликтов приоритет всегда имеет текущая TSL-структура, что вполне естественно.

Загрузка данных выполняется в два этапа. Сначала из набора данных извлекается имя TSL-проекта-шаблона (которое хранится в нем), проект с этим именем в своем текущем виде загружается из дисковых TSL-файлов модулей и в памяти формируется начальный образ данных. Затем выполняется загрузка фактической структуры и значений данных, записанных в TDF-наборе, которые "накладываются" на имеющуюся к этому времени в памяти начальную структуру. В целом механизм хранения и загрузки фактических данных обеспечивает их устойчивость к изменениям TSL-структуры.

Доступ к данным

Данные образуют в памяти логическую древовидную структуру вложенных объектов-записей с единственным корневым объектом, причем каждый объект-запись представляет собой список именованных объектов-полей. Поле, в свою очередь, является массивом (ранга 0, 1 или 2) элементов данных соответствующего типа, заданного в проекте структуры. Доступ к данным в TSS3 осуществляется на основе *контекста*.

Контекст — это один из объектов-записей в составе общей структуры, являющийся логически текущим в любой момент работы с данными и выполняющий при этом двоякую роль:

- поля в его составе доступны для операций программного чтения и записи данных. Операции при этом адресуются к полям по их именам;
- контекст является логической единицей интерфейса редактирования, который строится программными средствами TSS3 как таблица (список) его полей. Для редактирования данных каждого поля используется соответствующий его типу визуальный элемент.

Таким образом, непосредственный доступ к любым данным (локализованным в полях объектов-записей) возможен в пределах текущего контекста структуры. Этот контекст должен быть предварительно установлен, для чего в TSS3 предусмотрены операции перехода (смены контекста):

- к вложенному объекту-записи. Указываются имя поля в текущем контексте (оно должно иметь тип записи) и соответствующее число индексов (0, 1 или 2) в массиве данных поля;
- к вмещающему объекту;
- к корневому контексту (root).

Состав оболочки TSS3

Состав оболочки TSS3, потоки данных и взаимодействие с Фортран-программой схематически представлены на рис. 2.

Программная оболочка TSS3 включает следующие компоненты — исполняемые модули:

- программу TSS3P для описания структуры данных;
- программу TSS3D автономного редактирования наборов данных;
- интерфейсную оболочку TSS3S — программный сервер, исполняющий запросы Фортран-программы (клиента) в процессе ее выполнения (runtime server).

Все программы оболочки TSS3 разработаны в среде Borland Delphi версии 7.0.

Следует подчеркнуть важность наличия в составе TSS3 средств автономного редактирования наборов данных. С их помощью становится возможным пакетный режим расчетов, когда наборы входных данных подготавливаются предварительно, а запуск счетных программ и ввод данных осуществляются полностью автоматически, без диалога с оператором.

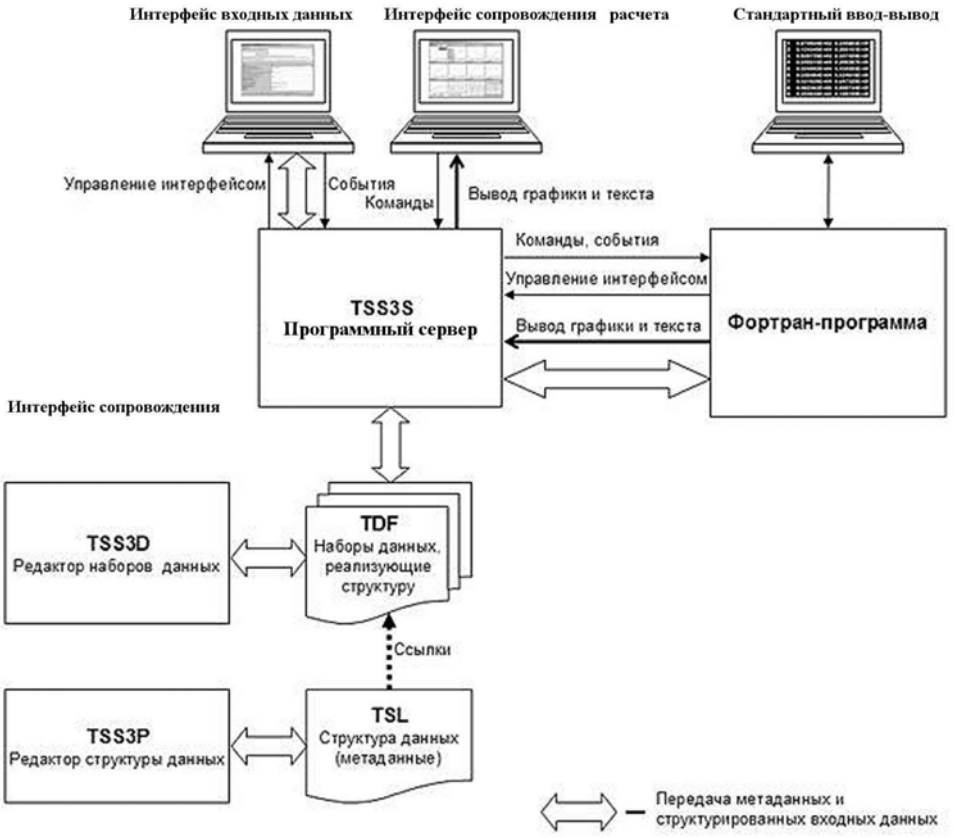

Рис. 2. Взаимосвязи компонентов комплекса TSS3

Программа описания структуры данных TSS3P

Для описания и редактирования структуры данных в составе оболочки TSS3 разработана программа TSS3P (TSS3 Project editor). Она предоставляет визуальную среду проектирования структуры данных — определения пользовательских типов (перечислений, множеств и записей) в составе модуля, а также построения многомодульных проектов. При этом иерархия вложенности записей отображается графически в виде дерева. Программа осуществляет чтение и запись модулей на языке TSL.

Программа автономного редактирования наборов данных TSS3D

Собственно создание наборов входных данных по описанной структуре и их редактирование пользователем может осуществляться либо предварительно, либо уже на стадии выполнения счетной Фортран-программы (клиента) при взаимодействии с сервером. В первом случае имеется по существу пакетный режим счета и функции сервера заключаются только в загрузке данных из уже подготовленного набора и их передаче в программу-клиент. Во втором случае сервер обеспечивает также необходимый интерфейс пользователя для редактирования данных.

Для обеспечения пакетного режима счета, а также "заготовки" типовых шаблонов наборов данных необходимы средства работы с TDF-наборами в автономном режиме. Такие средства для оболочки TSS3 содержатся в программе TSS3D (TSS3 Data editor). Эта программа предоставляет визуальную среду редактирования данных в TDF-наборах.

Программный сервер TSS3S

Центральным компонентом в составе комплекса TSS3 является программный сервер TSS3S. Этот модуль реализует все функции интерфейсной оболочки Фортран-программы — клиента. Главной его задачей является поддержка пользовательского интерфейса для задания входных данных на основе их структуры и обмен этими

данными с программой-клиентом. Кроме того, сервер TSS3S решает ряд практически актуальных дополнительных задач:

- ввод *внеструктурных* данных;
- доступ к стандартным системным диалогам;
- сопровождение счета, в том числе:
  - вывод текстовой информации;
  - вывод графиков одномерных зависимостей;
  - индикация хода выполнения расчета;
  - управление счетом через меню, формируемое пользователем;
- различный низкоуровневый программный сервис.

Программно сервер реализован в виде динамически загружаемой библиотеки TSS3S.DLL, код которой выполняется в едином адресном пространстве с Фортран-программой (клиентом). При взаимодействии клиента и сервера данные и управление передаются в обоих направлениях (механизм функций обратного вызова). Библиотека TSS3S.DLL экспортирует более 130 функций.

Программный интерфейс

Программный интерфейс сервера TSS3S выполнен в виде единого модуля на языке Фортран-90. Следует отметить принципиально возможную зависимость программного интерфейса от используемой реализации языка Фортран-90, где может различаться синтаксис директив компилятора (не регулируется стандартом языка). В настоящее время модуль интерфейса является единым для систем разработки Compaq Visual Fortran 6.6 и Intel Fortran 10 в среде 32-разрядных ОС Microsoft Windows.

При реализации модуля интерфейса TSS3S применялись некоторые общие принципы самодокументированности, направленные на облегчение использования TSS3 в прикладных Фортран-программах и предотвращение семантических ошибок:

- полное и явное указание атрибутов формальных параметров интерфейсных функций в их прототипах (отказ от соглашений по умолчанию языка Фортран);
- стандартизация символических имен функций и именованных констант, а также имен формальных параметров функций;
- широкое использование механизма перегрузки функций (overload) в языке Фортран-90.

При проектировании программного интерфейса TSS3 особое внимание уделялось его компактности, т. е. сокращению числа интерфейсных функций. Эта цель была достигнута за счет объединения логически однородных процедур в единую точку входа в библиотеку (экспортируемую процедуру), параметризованную входным параметром — кодом операции, а также перегрузки имен функций.

Обзор функций программного интерфейса

Всю совокупность функций модуля TSS3 можно разбить на ряд логических групп, схематически представленных на рис. 3.

Пользовательский интерфейс реализуется оболочкой TSS3 через множество окон различных типов. Структура окон TSS3 показана на рис. 4. Как видно из рисунка, сервер TSS3 имеет два окна верхнего уровня: главное окно редактирования структурных данных соответствует стадии задания входных данных, главное окно сопровождения счета является стандартным в ОС Windows MDI-контейнером, содержащим пользовательское меню и произвольное число окон вывода текста и графиков. Имеется также ряд специализированных модальных окон, реализующих *низкоуровневый* ввод внеструктурных данных, выбор объектов файловой системы (файлов и каталогов) и вывод текстовых сообщений.

На рис. 4 отражен также общий принцип редактирования составных объектов структурных данных: вложенные в текущий контекст составные объекты-массивы или записи редактируются во вложенных модальных окнах.

Состав группы полей в текущем контексте, выводимой на экран для редактирования, определяется либо списком имен, либо битовой маской групповой принадлежности (код группы — статический целочисленный атрибут поля, определяемый в структуре).

Рассмотрим подробнее основные принципы построения интерфейса редактирования. В его основе всегда лежит таблица, структура которой определяется структурой редактируемых данных — набором редактируемых одновременно полей, их типом данных и размерностью массива. Зависимость выбираемого формата таблицы от этих факторов показана на рис. 5.

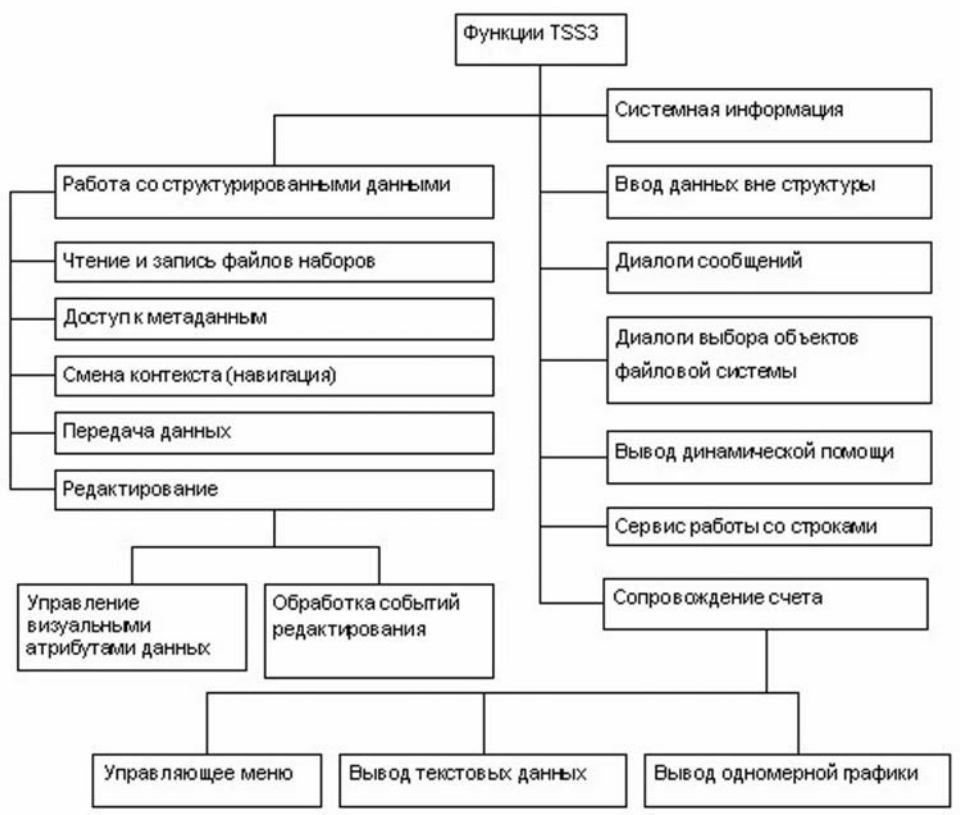

Рис. 3. Группы функций программного интерфейса TSS3

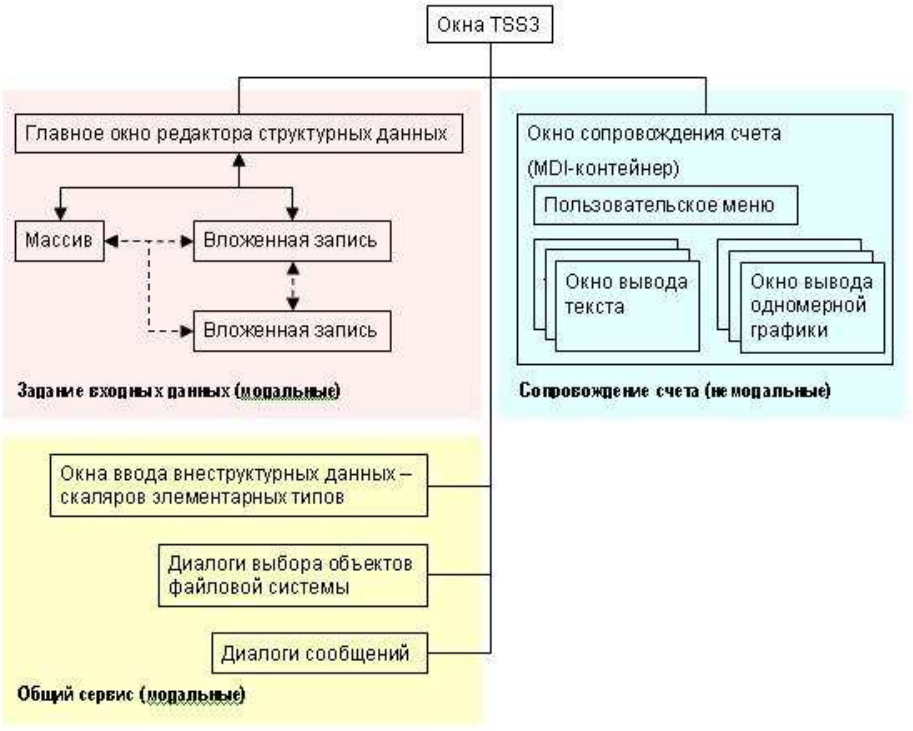

Рис. 4. Типы окон пользовательского интерфейса TSS3

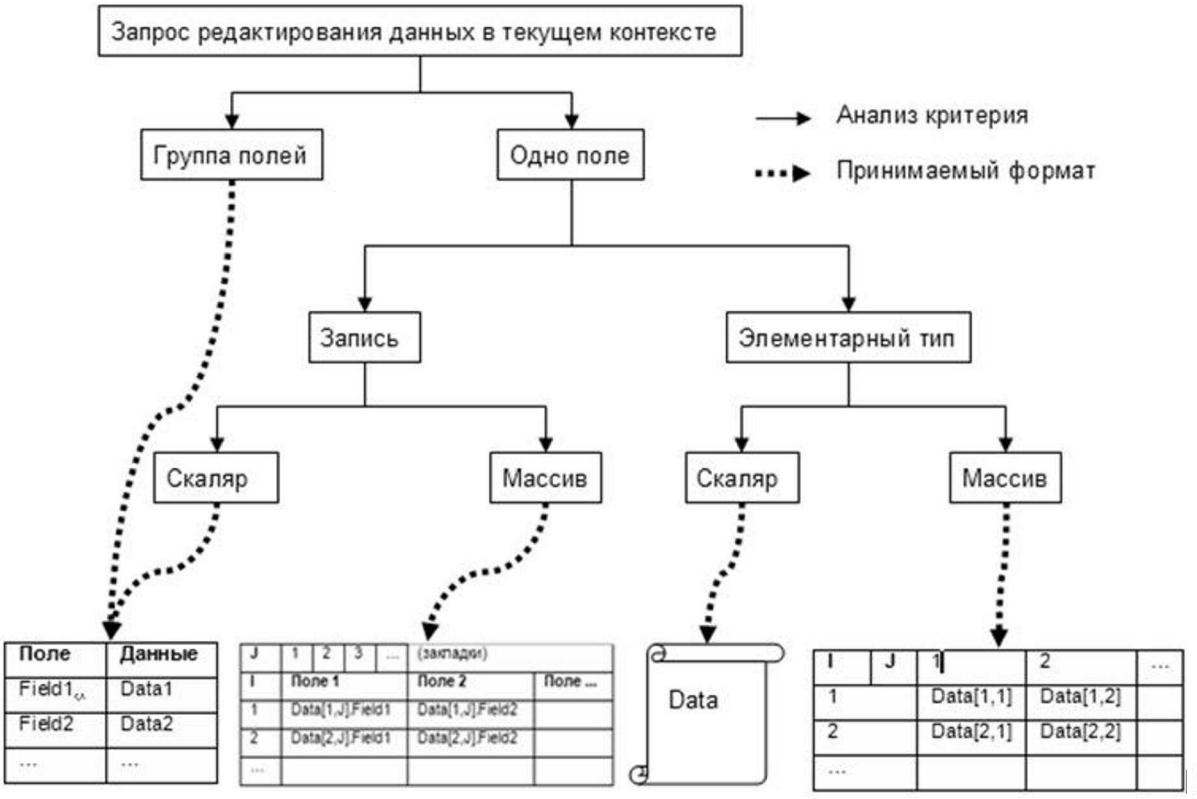

Рис. 5. Выбор структуры таблицы редактирования данных в TSS3

Для элементарных типов в ячейках таблиц выводится строковое представление данных, для записей — компактная текстовая форма вида

поле1=значение1, поле2=значение2, ...

Редактирование величин числовых и строковых типов осуществляется непосредственно в ячейках таблиц, типов-селекторов — в выпадающих списках, динамически создаваемых на экране при входе в ячейку. Если объект в ячейке таблицы является массивом или записью, ячейка содержит кнопку для вызова редактора данных во вложенном модальном окне. Структура таблиц во вложенных окнах также определяется по алгоритму, показанному на рис. 5. Визуальное представление данных в таблице определяется статическими и динамическими (устанавливаемыми программно в клиент-коде) атрибутами полей.

Вид главного окна сервера TSS3S (режим редактирования группы полей) представлен на рис. 6. Основная область окна — таблица редактирования данных. Вместо имен полей в таблице используются их информативные описания, заданные в TSL-структуре. Уровень доступа (видимости) полей задается трехпозиционным селектором Basic/Common/Expert (структурный атрибут поля) в правом нижнем углу главного окна. Верхняя часть окна имеет область для отображения динамической контекстной подсказки. Имеется возможность форматирования текста подсказки.

Обработка событий

Значительно повысить информативность и эргономику пользовательского интерфейса для задания входных данных в оболочке TSS3 позволяет механизм обратного вызова из оболочки Фортран-программы для обработки событий редактирования. После запуска редактирования вызовом соответствующей функции программного интерфейса в оболочке автоматически строится таблица представления данных и выполняется переход в модальный режим их редактирования. Выход из функции редактирования и возврат управления программе-клиенту происходит при нажатии экранных кнопок, каждой из которых приписан собственный целочисленный код — модальный результат функции редактирования. Наличие механизма обратного вы-

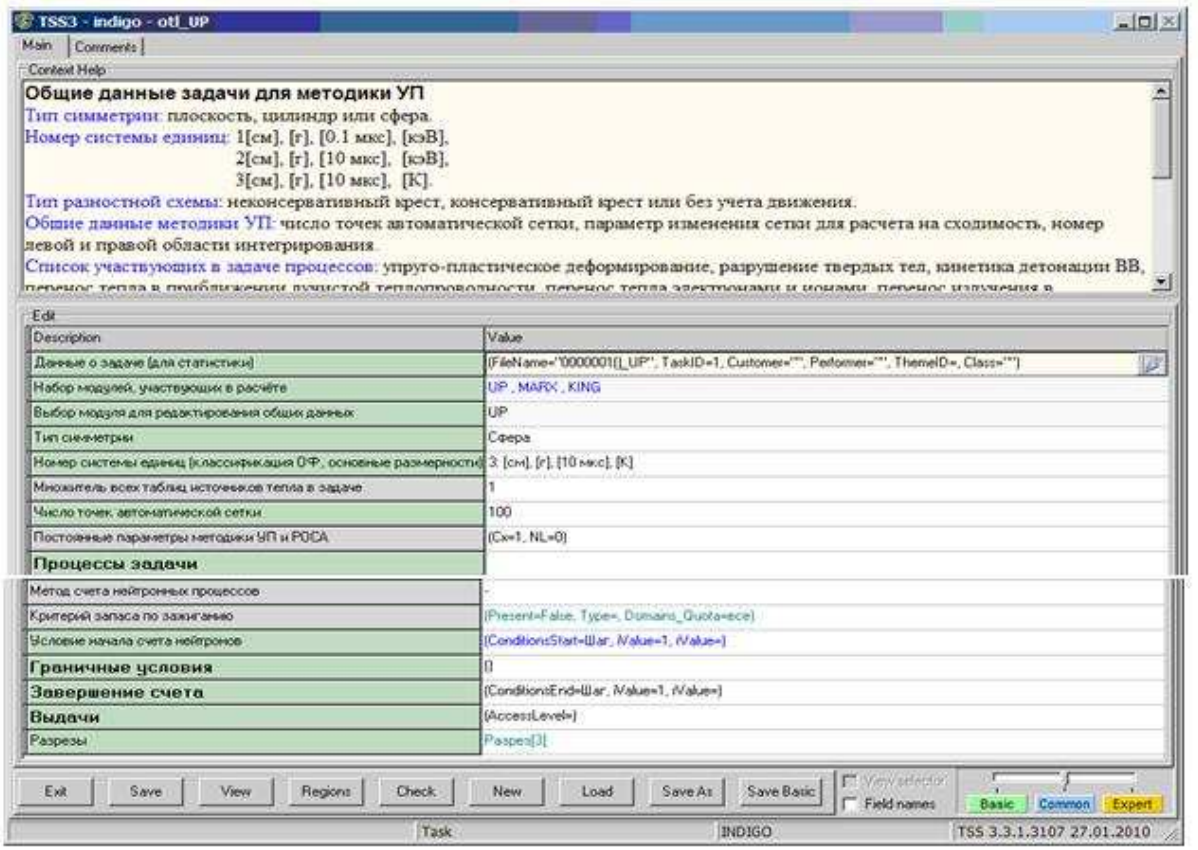

Рис. 6. Вид главного окна редактирования сервера TSS3S (группа полей)

зова Фортран-программы из оболочки во время редактирования дает принципиальную возможность гибкого управления как данными (чтение, анализ, предзасылка), так и их визуальным представлением в зависимости от текущей ситуации редактирования.

На этапе редактирования входных данных оболочка TSS3 поддерживает три разновидности событий:

- команды, определенные пользователем (ассоциированы с элементами интерфейса — кнопками, меню);
- события-извещения об изменении состояния редактирования (навигация в структуре данных и модификация значений);
- изменение состояния объектов типа множества (set).

Все процедуры-обработчики в Фортран-программе, вызываемые из оболочки во время редактирования, имеют полный доступ к данным и их визуальным атрибутам.

Дополнительный сервис

Программный сервис оболочки TSS3 включает также ряд функций общего назначения, облегчающих решение типовых задач:

- ввод скалярных внеструктурных данных — чисел и строк — через отдельные модальные окна;
- работа со строками (преобразование регистра, разбивка на лексемы);
- информационные функции (получение имени компьютера, пользователя);
- вывод диагностических сообщений;
- выбор объектов файловой системы (файлов и каталогов).

Сопровождение счета

Оболочка TSS3 предоставляет Фортран-программе (счетному программному комплексу) сервис по организации пользовательского интерфейса и на стадии счета:

– построение пользовательского меню и обработка его событий в процедуре-обработчике Фортран-программы;
– индикацию хода выполнения расчета;
– вывод текста в аналог консоли;
– динамический вывод графиков одномерных зависимостей.

Этот интерфейс реализуется с помощью второго главного окна — MDI-контейнера, содержащего пользовательское меню управления счетом и произвольное число одновременно открытых окон текста и графики. Разумеется, при этом остаются доступными и все остальные функции TSS3, условно относимые к вводу (структурные и внеструктурные данные, диалоги, общий сервис).

Пример окна сопровождения счета представлен на рис. 7.

Окно сопровождения счета включает меню для интерактивного управления счетом задачи. При выборе его пунктов пользователем из оболочки вызывается указанная процедура-обработчик Фортран-программы и происходит передача ей кода выбранного пункта меню и некоторых дополнительных параметров.

Заключение

Первая реализация программной оболочки TSS была выполнена в среде MS-DOS в 1994 г. [4]. Ее концепция была намного более ограниченной: не допускалась вложенность объектов-записей, а древовидная иерархия структуры задавалась непосредственно в виде графа отношений *родитель — потомок*. Кроме того, отсутствовали многие из визуальных атрибутов данных и механизм событий редактирования, т. е. создаваемые на основе TSS интерфейсы были менее информативны и весьма статичны.

Рассмотренная в статье версия программной оболочки TSS — TSS3 — создана в 2003 г. и развивается до сих пор. С одной стороны, она отражает прогресс в описании структуры данных и программной реализации компонентов TSS в среде Windows, а с другой, решает ряд новых задач сервисного обслуживания счетных программ на стадии

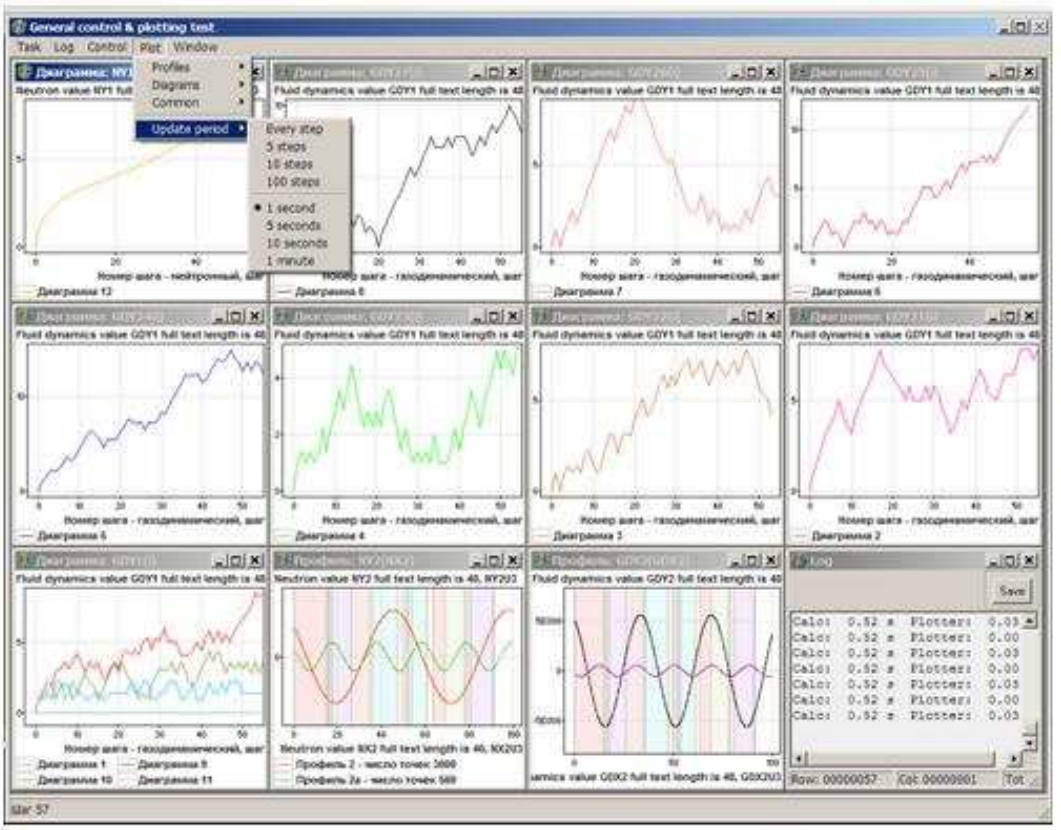

Рис. 7. Окно сопровождения счета оболочки TSS3

собственно счета. Современная версия оболочки TSS3 позволяет автоматически строить пользовательские интерфейсы прикладных программных комплексов на языке Фортран на основе декларативного описания структур входных данных. В сочетании с поддерживаемыми TSS3 механизмами динамического межпрограммного взаимодействия получаемые интерфейсы обладают достаточно высоким уровнем информативности и эргономики. Обеспечивается также поддержка пользовательского интерфейса на стадии счета с динамическим выводом графиков одномерных зависимостей.

Оболочка TSS3 нашла применение в ряде программ, среди которых можно отметить систему константного обеспечения нейтронно-физических расчетов [5].

Список литературы

1. *Бартеньев О. В.* Современный Фортран. М.: Диалог — МИФИ, 1998.
2. *Архангельский А. Я.* Программирование в Delphi для Windows 2006, 2007. М.: Бином-Пресс, 2010.
3. *Вайсфельд М.* Объектно-ориентированный подход: Java, .NET, C++. М.: Кудиц-образ, 2005.
4. *Гребенников А. Н., Крутько Н. А., Мжачих С. В.* Программная оболочка входных данных TSS // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 1997. Вып. 3. С. 10—17.
5. *Гребенников А. Н., Иванникова В. Н., Крутько Н. А., Фарафонтов Г. Г.* Современное состояние системы константного обеспечения нейтронно-физических расчетов // Там же. Вып. 4. С. 76—82.

Статья поступила в редакцию 18.10.10.