УДК 004.415.2

05.13.00 Информатика, вычислительная техника и управление

ТРЕХЗВЕННАЯ КЛИЕНТ-СЕРВЕРНАЯ ИНФОРМАЦИОННАЯ СИСТЕМА ПО КОНФИГУРИРОВАНИЮ БАЗ ДАННЫХ

Нестеров Геннадий Дмитриевич
к.т.н., профессор
e-mail:nnnnnnn46@ mail.ru
SPIN код = 5921-3711
Академия маркетинга и социально-информационных технологий (ИМСИТ), Краснодар, Россия

В настоящее время актуальна задача использования технологии «клиент-сервер» для баз данных. К ее достоинствам относятся высокая степень безопасности, возможность работы с мультимедийными и нестандартными данными, нетребовательность к аппаратной мощности клиентских станций, территориальная независимость, работа как в локальных, так и в глобальных сетях. В статье дана концепция и описана реализация клиент-серверной информационной системы по конфигурированию баз данных на основе плагинов, позволяющих гибко, быстро и эффективно вносить изменения в существующее приложение по работе с базами данных. Именно этот подход решает проблему использований подключения одной программы к нескольким СУБД

Ключевые слова: «КЛИЕНТ-СЕРВЕР», ПЛАГИН, КОНФИГУРИРОВАНИЕ, ПРОЦЕДУРЫ, ИНТЕРФЕЙС, КЛАСС, МОДУЛЬ

UDC 004.415.2

Information, calculation technology and management

THREE-TIER CLIENT-SERVER INFORMATION SYSTEM OF CONFIGURING DATABASES

Nesterov Gennadiy Dmitrievich
Cand.Tech.Sci., professor
e-mail:nnnnnnn46@ mail.ru
RSCI SPIN code =5921-3711
Academy of Marketing and Social-Information Technologies (IMSIT), Krasnodar, Russia

Currently, the problem of using a "client-server" technology for database is up-to-date. Among its advantages it has a high degree of security, the ability to work with multimedia and non-standard data, simple tastes to the hardware power client stations, the territorial independence of work in both local and wide area networks. The article presents the concept and describes the implementation of a client-server information system configuration databases based on plug-ins that enable flexible, fast and efficient to make changes to an existing application for working with databases. This approach solves the problem of using one program to connect to multiple databases

Keywords: CLIENT-SERVER, PLUG-IN, CONFIGURING, PROCEDURES, INTERFACE, CLASS, MODULE

В настоящее время в силу ряда преимуществ получила распространение технология построения баз данных - «клиент-сервер» [1]. К ее достоинствам относятся высокая степень безопасности, возможность работы с мультимедийными и нестандартными данными, нетребовательность к аппаратной мощности клиентских станций, территориальная независимость, работа как в локальных, так и в глобальных сетях.

В связи с этим многие организации ведут работы в направлении переноса всех программ, использующих технологию предыдущего поколения «файл-сервер», на технологию «клиент-сервер».

Поэтому возникла задача написания клиент-серверной информационной системы по конфигурированию баз данных.

Для ее реализации применена система конфигурирования на основе плагинов, позволяющих гибко, быстро и эффективно вносить изменения в существующее приложение по работе с базами данных. Именно этот подход решает проблему использований подключения одной программы к нескольким СУБД.

Определены следующие технологии подключения к базам данных: dbExpress, представляющая собой архитектуру создания драйверов данных (разработка Embarcadero) и заменяющая устаревший BDE . Ее применение позволяет:

- уменьшить затрачиваемые ресурсы;
- обеспечить наибольшую скорость работы;
- реализовать кросс-платформенность;
- достаточно просто разрабатывать драйверы;
- пользоваться расширенными возможностями управления трафиком и памятью.

Отличительной чертой драйверов dbExpress являются высокая скорость работы и маленький размер, поскольку они обеспечивают небольшую функциональность.

Для доступа к данным используется ADO — интерфейс программирования приложений, который основан на технологии компонентов ActiveX. Следует отметить, что ADO разрешает представлять данные из разнообразных источников (в объектно-ориентированном виде).

На основе этих технологий реализованы расширения. Каждое расширение снабжено специализированным сервером приложений, реализующим бизнес-логику и определяющим изоляцию транзакций.

В каждом плагине реализовано промежуточное (третье звено) подключение к базе данных. При работе приложения применяется протокол

передачи данных – DCOM, который использует технологию COM. Технология COM (Component Object Technology) представляет собой объектно-ориентированную программную спецификацию, предназначенную для повышения надежности взаимодействия компьютерных программ между собой. COM-объекты прозрачно друг для друга модифицируются, поскольку через GUID обеспечивается доступ к объектам. В COM технологию включены библиотека с набором стандартных интерфейсов, определяющих ядро функциональности COM и некоторый набор API функций, для создания COM-объектов и управления ими. Так как COM является стандартным средством операционной системы Windows, на фазе разработки не понадобится формирование специализированных алгоритмов для ее реализации

Были установлены бизнес, пользовательские и функциональные требования к разрабатываемой системе.

Бизнес- требования определяют будущую информационную систему как неотъемлемую часть информационной структуры предприятия, позволяющей автоматизировать конфигурирование баз данных посредством SQL скриптов с возможностью создания некоторых информационных артефактов, которые могут выполняться во взаимодействии с другими системами. Эти артефакты должны быть реализованы в виде динамических библиотек, обладающих возможностью определять очередь их выполнения, выстраивая таким образом «цепочки» обновления метаданных на объектах обслуживания.

Пользовательские требования заключаются в создании наиболее удобных механизмов взаимодействия с конечным пользователем.

Функциональные требования связаны с указанными двумя первыми, вследствие чего были созданы:

- модули-контейнеры как некоторая среда выполнения плагинов, в которой будут заложены основные элементы интерфейса и базового функционала;

- динамически компонуемые библиотеки, используемые в качестве расширений.

Модули-контейнеры представляют собой совокупность модулей, При их компиляции создается обычный файл с расширением EXE, в котором заложен базовый функционал информационной системы, а также определен механизм загрузки и подготовка к выполнению расширений. Следует заметить, что именно в этих модулях заложен механизм взаимодействия информационной системы с пользователем.

Разработка выполнена в два основных этапа:

- создание непосредственно программной оболочки и ее расширений для взаимодействия с базой данных;
- создание конфигурационных процедур и функций в базе данных объекта обслуживания.

На втором этапе определены необходимые хранимые процедуры и триггеры базы данных для вызова их из модулей расширений.

Архитектурное представление разрабатываемой системы можно представить в следующем виде.

Логическое представление. Разрабатываемая система в логике своей работы использует как стандартный функционал среды разработки, так и специально разработанное ядро плагинной системы. Стандартный функционал включает в себя методы структурного и объектно-ориентированного программирования, базовые методы манипулирования строками, целыми и дробными числами, определение условных и циклических конструкций. Ядро плагинной системы определяет хранилище адресов памяти, в которые загружены библиотеки плагинов, и вызывает их выполнение по запросу пользователя.

Процедурное представление. В качестве обработки данных используются процедуры и функции, цель которых «разгрузить» основной код программы, а также облегчить последующее сопровождение.

Реализационное представление включает в себя подготовку приемочного тестирования на стороне заказчика, а также формирование тест-карты.

В разрабатываемом приложении реализована плагинная система на основе интерфейсов. Реализуемые интерфейсы классов приложения и плагинов передаются как ссылки друг другу для управления различными процессами и распределения потоков данных. С этой целью создан отдельный модуль, в котором имеются описания реализуемых интерфейсов. Он используется как самим приложением, так и библиотекой расширения, что в свою очередь определяет однозначность объявления интерфейсных методов в классах.

Интерфейсы выполняют следующие функции:

- обработка переданного текста (в плагине);
- отображение статуса выполнения плагина (приложение);
- отображение хода выполнения соответствующего плагина (приложения).

Соответственно плагин получает ссылки на интерфейсы приложения, чтобы сообщать о ходе своего выполнения. Приложение, в свою очередь, получает ссылку на интерфейс выполнения плагина для запуска обработки конфигурационного файла. В целях определения загрузки и выполнения плагина в основном приложении создано, так называемое, ядро плагинной системы, которое реализует всю логику выполнения плагина.

Ядро плагинной системы состоит из двух классов. Первый - представляет собой объект, хранящий информацию о плагине, методах его реализации и реализующий интерфейсные методы, которые будут переданы плагину в момент его инициализации. Второй класс является некоторым хранилищем классов плагина, загружая их из переданной директории для доступа к ним из основного приложения.

Выполнение плагинов осуществлено следующим образом:

- вызов зарезервированной экспортируемой функции из плагина, которая передает имя плагина, его гуид и ссылку на реализуемый интерфейс выполнения и получает ссылки на интерфейсы приложения;
- передача интерфейсному методу плагина в качестве параметра текста конфигурации;
- вызов интерфейсного метода плагина;
- вызов плагином интерфейсных методов приложения для информирования пользователя о ходе своей работы.

Такой подход не только универсален, но и удобен в определении минимального кода для реализации описанного механизма.

На стороне плагина должна быть описана некоторая процедура определенной структуры, которая принимает в качестве входных параметров ссылки на интерфейсные методы основного приложения и возвращает имя плагина, гуид и ссылку на метод выполнения. Таким образом приложения и плагин беспрепятственно «общаются» между собой.

Основное неудобство данного метода заключается в объявлении экспортируемой процедуры. Об этом должны «знать» как приложение, так и плагин. Решение этой проблемы было определено как создание директивы описания в общем интерфейсном модуле, где разработчик плагина сможет посмотреть объявление данной функции и включить ее реализацию в программный код плагина. Гуид плагина – это некоторый уникальный идентификатор, созданный, чтобы определить индексации в ядре плагинной системы, в которой доступ к выполнению плагина будет осуществляться непосредственно по переданному гуиду. Вызов на выполнение некоторого плагина реализован в методе класса хранилища плагинов, который в качестве входного параметра будет получать имя плагина и его гуид. Далее реализуется поиск в массиве указателей на класс плагинов. При нахождении необходимого вызывается соответствующий метод класса плагина на его выполнение.

По окончании работы программы вызываются деструкторы классов ядра плагинной системы во избежание ошибок доступа к памяти.

С целью реализации описанного функционала основного приложения разработаны 6 программных модулей, взаимодействие между которыми организовано путем подключения к пространству имен названия файла модуля без расширения.

Указанные модули удовлетворяют условиям:

1. Размеры модуля не должны быть слишком маленькими или большими.

Если модули малы, то модульная структура программы громоздкая и накладные расходы по ее оформлению могут не окупиться. Использование больших модулей неудобно в отношении изучения и внесения изменений из-за возможности существенного увеличения общего времени повторных трансляций программы в процессе ее отладки.

2. Модуль должен быть функционально прочным.

Функционально прочный модуль выполняет одну определенную функцию. Причем, в процессе ее реализации им могут использоваться и другие модули.

3. Модуль должен быть информационно прочным.

Информационно прочный модуль выполняет несколько операций (функций) над одной и той же структурой данных, неизвестной вне его. Такой модуль содержит вход и форму обращения к нему для каждой из этих операций.

Разработанная программа использует следующие стандартные модули. *Windows* – работа со стандартными библиотеками Windows; *Messages* – обработка сообщения Windows; *SysUtils* – отвечает за шрифты и математические операции; *Classes* – поддержка классов; *Graphics* – графические фильтры; *Controls* – элементы управления; *Forms* – работа с формами; *Shellapi* – модуль для работы с api–функциями; *System* – системные проце-

дуры и функции; *Dialogs* – вывод диалогов пользователей; *Variants* – работа с вариантными типами данных.

Были разработаны модули:

MainFormUnit.pas. Предназначен для взаимодействия с остальными модулями программы. Реализует главную форму, включающую главное горизонтальное меню и панель инструментов, лог выполнения плагинов, список плагинов а также средства визуализации данных.

ChildFormUnit – модуль дочерней формы интерфейса MDI. Определяет отображение текста конфигурационного файла, содержит набор компонентов прорисовки лексхем.

ConfigFormUnit – модуль формы конфигурации программы. Реализует функционал редактирования директорий загрузки плагинов и шаблонов, позволяет назначать горячие клавиши к основным функциям программы.

ConfigSynUnit – модуль конфигурирования лексхем. Позволяет загружать файл лексхемы для последующего редактирования.

ShablonUnit – модуль формирования шаблонов. Определяет механизмы отображения списка шаблонов, переименования, удаления и сортировки.

CorePlugin – модуль ядра плагинной системы. Реализует два класса плагинной системы.

Для взаимодействия плагина и приложения создан модуль реализуемых интерфейсов, включающий в себя:

- интерфейс обратного вызова для ведения лога и процента выполнения плагина;
- интерфейс обратного вызова, возвращающий обработанный файл конфигурации для выполнения другими плагинами;
- описание формата экспортируемой функции, возвращающее имя, тип, гуид и ссылку на интерфейс выполнения плагина;
- интерфейс выполнения плагина.

На основе описанных программных модулей созданы соответствующие элементы интерфейсов.

Основные элементы главного окна: главное меню, панель инструментов, рабочая область, панель плагинов, панель лога работы плагинов.

Окно настроек программы имеет две вкладки. Первая содержит элементы, отображающие общие настройки приложения. Вторая служит для конфигурирования горячих клавиш.

Окно конфигурирования лексхем разделено на следующие структурные элементы:

- Лексхема – содержит выпадающий список, отображающий ее название.
- Параметры – включает в себя несколько областей, сгруппированных по смысловому значению, а именно: идентификаторы – задает настройку прорисовки ключевых слов; комментарии – задает настройку прорисовки комментариев; символы – задает настройку прорисовки символов; строки – задает настройку прорисовки строковых выражений; настройки редактора – определяет параметры редактора, в который загружается файл конфигурирования базы данных; разделители – определяет разделители при множественном выделении; операторы вложений – позволяет создавать массив операторов, которые будут определены как вложения.
- Окно конфигурирования шаблонов реализует функционал модуля конфигурирования шаблонов. Служит для создания шаблонов типовых конфигураций баз данных. Содержит элементы интерфейса для отображения списка зарегистрированных ранее шаблонов в базе данных (список), а также выполнения команд открытия, удаления, переименования, сортировки, удаления шаблонов и поиска шаблонов.

- Дочернее окно программы реализует MDI интерфейс приложения. Содержит многострочные поля ввода для отображения файла конфигурации, а также экспорта текста в формат RTF.

Было проведено тестирование разработанного программного продукта.

Следует отметить, что для обеспечения последующего сопровождения и повышения его эффективности выполнена процедура модерации текстов исходных кодов приложения. При испытаниях и приемочном тестировании отмечены высокая эффективность разработанной информационной системы, интуитивно-понятный пользовательский интерфейс и качество его прорисовки, а также гибкость настроек приложения.

Список литературы

Шибанов С.В. Основы разработки баз данных в клиент-серверных приложениях. - Пенза: Пензенский государственный университет, 2010 [Электронный ресурс] Режим доступа: http://www.twirpx.com/file/456348/

List of references

Shibanov S.V. Osnovy razrabotki baz dannyh v klient-servernyh prilozhenijah. - Penza: Penzenskij gosudarstvennyj universitet, 2010 [Jelektronnyj resurs] Rezhim dostupa: http://www.twirpx.com/file/456348/