УДК 28.17.19

ИНТЕГРАЦИЯ ИНТЕРНЕТ-МАГАЗИНА И СИСТЕМЫ УПРАВЛЕНИЯ ПРЕДПРИЯТИЕМ *1С* ПРИ ПОМОЩИ МОДУЛЕЙ ЕВРИПТЕРИД

© О.В. Крючин, К.Р. Хабирова, Е.В. Вязовова

Ключевые слова: интернет-магазины; система управления предприятием; протокол передачи данных.
Описана технология интеграции системы управления предприятием *1С* и интернет-магазинов, базирующихся на различных движках. Приводится подробное описание архитектуры системы и особенности ее технической реализации.

ВВЕДЕНИЕ

Традиционно в России управление торговым предприятием (магазин и т. п.) в большинстве случаев осуществляется при помощи программ *1С*, занимающих в настоящее время более 75 % рынка. В то же время стремительно возрастает значение интернет-магазинов, число которых пока уступает аналогичному на Западе, но постепенно приближается к нему. Таким образом, интеграция интернет-магазинов и *1С* приобретает все большую актуальность. Программистами *1С* создано несколько решений, но все они требуют существенной доработки.

Исходя из вышесказанного, целью данной работы является разработка системы, позволяющей интегрировать *1С* версии 7 «*Торговля и склад*» с интернет-магазинами, основанными на движке *EuripterideShop*.

СТРУКТУРА СИСТЕМЫ

Компоненты системы. Разрабатываемая система представляет собой совокупность нескольких компонентов, архитектура которых представлена на рис. 1:

– *1С* (произведено изменение конфигурации и добавлено несколько внешних обработок);
– *dll* (содержит вызываемые из *1С* функции работающие с СУБД);
– *mysql* (СУБД, содержащую контент, необходимый для интернет-магазина);
– *web* (написанный на *PHP* интернет-магазин).

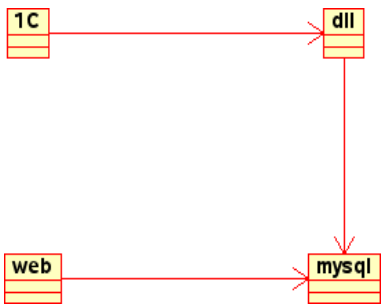

Рис. 1. Диаграмма компонентов системы

Схема взаимодействия. При изменении базы *1С* пользователем (*vender*) происходит вызов *1С*-функций, которые в свою очередь вызывают функции *dll*, вносящие изменения в СУБД. При запросах пользователя (*client*) веб-компонент (интернет-магазин) обращается к СУБД (рис. 2).

Программно-аппаратная архитектура. Структура распределения компонентов по аппаратным ресурсам показана на рис. 3. Как можно видеть, система состоит из трех частей (подсистем) – *Vendor*, *Server* и *Client* (из которых первые две входят в состав дистрибутива).

Компоненты системы *Vender* работают по следующему алгоритму:

1) пользователь изменяет контент в базе *1С*;
2) *1С* обращается к набору *dll*;
3) *dll* обращается к удаленной СУБД.

ПОДСИСТЕМА VENDOR

Механизм. Для реализации подсистемы *Vendor* в конфигурацию *1С* внесены следующие изменения:

1) изменен справочник «*Номенклатура*» – изменен код элемента на числовой и добавлены реквизиты «*Картинка*» и «*КартинкаБольшая*»;
2) изменена форма элемента – добавлены элементы отображения картинок;
3) изменен модуль элемента – добавлены процедуры «*ВыборКартинки*» и «*ВыборБольшойКартинки*», в процедуру *ПриОткрытии* вставлены открытия изображении, в процедуру *ПриЗаписи* – вызов процедуры *евпДобавлениеНоменклатурыИМ;*
4) изменена форма группы – добавлен элемент отображения картинок;
5) изменен модуль группы – добавлена процедура *ВыборКартинки,* в процедуру *ПриОткрытии* вставлены

Рис. 2. Схема взаимодействия

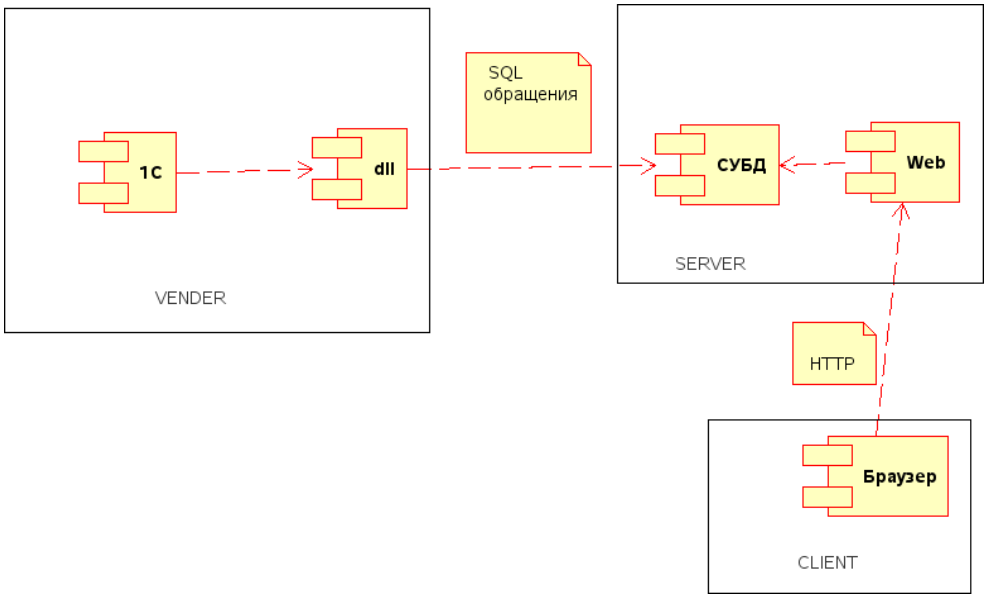

Рис. 3. Схема разделения компонентов по машинам

код открытия изображении, а в процедуру *ПриЗаписи* – вызов процедуры *евпДобавлениеГруппы НоменклатурыИМ* (табл. 1). Глобальный модуль содержит ряд функций (*s*-методов), обращающихся к *dll* и вызывающих оттуда *d*-методы. *S*-методы предназначены для внесения изменений в СУБД при обновлении справочников *1С*. Соответственно, вызываются они именно из модулей справочников (также эти функции вызываются при синхронизации базы).

Логика выгрузки. Для выгрузки номенклатуры в *DLL* предназначены два метода:

1) *евпСинхронизацияНоменклатуры*;
2) *евпВыгрузкаНоменклатуры*.

Первый из этих методов добавляет в СУБД данные из справочников, а второй очищает соответствующие таблицы в базе и заносит туда новые данные (т. е. разница в этих методах заключается в том, что первый не удаляет старые данные из СУБД). Таким образом, первый метод последовательно для каждой строки из справочника номенклатур вызывает метод *евпДобавлениеНоменклатуры* / *евпДобавлениеГруппыНоменклатуры* (в зависимости от того, является ли данный элемент справочника группой). Метод *евпВыгрузкаНоменклатуры* осуществляет очищение таблиц, вызывает *d*-метод *clearNomenclature* СУБД, а затем метод *евпСинхронизацияНоменклатуры*.

Для выгрузки контрагентов предназначен метод *евпВыгрузкаКонтрагентов,* который очищает соответствующие таблицы в СУБД, а затем последовательно для каждой строки справочников *Контрагентов*, *Юридических Лиц*, *Физических Лиц*, *Банковских Счетов* и *Банков* вызывает соответствующую функцию добавления в СУБД.

Чтение заказов. При загрузке заказов из интернет-магазина в *1С* необходимо произвести чтение некоторых служебных справочников, в частности банков. Чтение производится следующим образом:

– последовательно вызывается каждая строчка из соответствующей таблицы в СУБД;

– происходит проверка в справочнике банков на наличие записи с соответствующем кодом (БИК);

– в случае отсутствия соответствующей записи в *1С*, информация о банке заносится в справочник.

Загрузка контрагентов в общем виде может быть представлена как чтение данных четырех типов: банковских счетов, юридических лиц, физических лиц и собственно контрагентов. Логика загрузки похожа на алгоритм чтения банков с одним исключением: в случае если при загрузке строки из СУБД оказывается, что запись с таким же кодом имеется в справочнике, происходит проверка на эквивалентность записей (эквивалентность в данном случае не подразумевает полного соответствия записей, записи считаются эквивалентными при идентичности нескольких основных реквизитов – для контрагента это код, наименование и т. д., второстепенные записи могут различаться – для контрагентов это *e-mail* и т. д.). Если записи не эквивалентны, то запись справочника переносится в конец (вычисляется максимальное значение кода, и значение на 1 больше максимального присваивается элементу), а на его место записывается элемент из СУБД.

Более логичным может показаться алгоритм, в котором элемент из СУБД записывается в конец справочника, а элемент, находившийся в справочнике до этого, остается неизменным. Этот алгоритм был бы более быстр по сравнению с текущим. Выбор используемого алгоритма обусловлен следующей причиной: элементы справочников *1С* связаны, а для элементов таблиц СУБД этого утверждать нельзя. Это значит, что при изменении кода элемента все ссылки изменяются автоматически, т. е. по-прежнему указывают на этот элемент. Поскольку реализация интернет-магазина произвольна, перемещать элементы в таблицах может быть затруднительно, и для упрощения синхронизации выбран алгоритм перемещения элементов справочника.

Алгоритм чтения заказов состоит из двух частей – чтения данных и создания документа. Первая часть – загрузка данных из внешней компоненты в таблицу значений содержит несколько шагов:

– создается таблица значений с полями: контрагент; номенклатура, цена, количество;

Таблица 1

Список методов подсистемы *Vendor* (*s*-методов)

Название	Параметры	Назначение
евпИнициализация	хост сервера с БД (ip-адрес), порт, имя БД, имя пользователя (логин), пароль, префикс таблиц, постфикс таблиц	загрузка внешней компоненты и вызов d-метода, осуществляющего подключение к СУБД
евпДобавлениеГруппыНоменклатуры	код, наименование, код родительской группы, комментарий	добавление или обновление группы номенклатуры с заданным кодом
евпДобавлениеГруппыНоменклатурыИм	код, наименование, код родительской группы, комментарий, имя фотографии	расширенное добавление или обновление группы номенклатуры с заданным кодом
евпДобавлениеНоменклатуры	код, наименование, код родительской группы, комментарий, полное наименование, артикул, остаток (обычно количество экземпляров на складе)	добавление или обновление номенклатуры с заданным кодом
евпДобавлениеНоменклатурыИм	код, наименование, код родительской группы, комментарий, полное наименование, артикул, имя фотографии, имя большой фотографии	расширенное добавление или обновление номенклатуры с заданным кодом
евпДобавлениеЦены	код номенклатуры, тип, процент, код валюты, значение	добавление или обновление цены номенклатуры с заданным кодом
евпДобавлениеТипаЦены	код, наименование, вычисляемость (1 – рассчитывается, 0 – задается), код базового типа (если выч=1), процент, код валюты, НДС, НП, комментарий	добавление или обновление типа цены с заданным кодом
евпУдалениеНоменклатуры	код	удаление выбранной номенклатуры
евпДобавлениеЮридическогоЛица	код, наименование, ИИН, ОКПО, юридический адрес, фактический адрес, телефон, полное наименование, комментарий	добавление или обновление юридического лица с выбранным кодом
евпДобавлениеБанковскогоСчета	код, наименование (тип), код банка, значение счета, код владельца	добавление или обновление банковского счета с выбранным кодом
евпДобавлениеБанка	код, наименование, кор. счет	добавление или обновление банка с выбранным кодом
евпДобавлениеКонтрагента	код, наименование, тип (1 – юр.лицо, 2 – физ.лицо), код юр.лица/физ.лица, код банковского счета, электронная почта, комментарий	добавление или обновление контрагента с выбранным кодом
евпУдалениеЭлемента	структура, содержащая информацию о выбранном элементе	установка для выбранного элемента пометки на удаление
евпВосстановлениеЭлемента	структура, содержащая информацию о выбранном элементе	снятие для выбранного элемента пометки на удаление

– последовательно считываются все данные о заказах, взятые из СУБД;

– эти данные заносятся в таблицу значений;

– после завершения загрузки таблица значений сортируется по реквизиту «*Контрагент*».

Вторая часть – создание документа и заполнение его табличной части – также содержит несколько шагов:

– последовательно берутся все строки таблицы значений, соответствующие конкретному контрагенту;

– для этих данных создается документ «*Неподтвержденная заявка*»;

– выбранный контрагент заносится в шапку;

– в шапку заносятся текущая фирма и валюта;

– свойству документа *ВидОперации* присваивается значение «*Перечисление.ВидыОперацийЗаявок.Неподтвержденная*»;

– вызывается функция глобального модуля *глНазваниеДокументаВЖурнале* и в качестве параметра передается объект документа;

– данные о номенклатуре, цене и количестве в табличную часть.

Политика. Для подключения к СУБД предназначена внешняя обработка *eurip_db_connect.ert*, которая содержит поля для ввода реквизитов подключения: хост (*ip*-адрес СУБД-сервера), порт (обычно 3306), имя пользователя (логин), пароль, имя схемы (базы данных), префикс таблиц, постфикс таблиц.

После ввода реквизитов пользователь нажимает кнопку подтверждения, и вызывается *s*-метод *евпИнициализация*. Таким образом, загрузка внешней компоненты происходит именно на требование пользователя подключиться к СУБД (вызов метода *евпИнициализация* происходит в предопределенном методе глобального модуля *ПриОткрытии*, а реквизиты подключения прописаны в ядре).

СИНХРОНИЗАЦИЯ

Назначение. Синхронизация подсистем *Server* и *Vender* осуществляет две важные задачи:

1) поддерживает актуальность товаров в СУБД; несмотря на то, что при действиях пользователя в *1С* изменения вносятся в СУБД в режиме реального времени, синхронизация необходима в силу того, что соединение *1С* с СУБД может быть не постоянным; в результате на какой-то момент обнаруживаются расхождения в справочниках *1С* и таблицах СУБД; синхронизация решает эту проблему;

2) обеспечивает занесение информации о заказах в *1С*.

Для обеспечения постоянной синхронизации с определенным периодом должны вызываться выгрузка номенклатуры и загрузка заказов.

Механизм. Для обеспечения синхронизации предназначена внешняя обработка *eurip_sinhro.ert,* которая содержит поля для ввода периодов выгрузки номенклатуры и загрузки заказов, а также поля для ввода интервалов запаздывания. При инициализации системы выгрузка номенклатуры и загрузка заказов будет произведена мгновенно после подтверждения установки значения и истечении интервалов задержки. Таким образом, если установить для выгрузки номенклатуры интервал задержки 0, период 10, а для загрузки заказов – 5 и 10, то номенклатура впервые будет выгружена сразу, второй раз – через 10 мин., третий – через 20 и т. д., а заказы будут загружены через 5, 15, 25, 35, ... мин.)

Алгоритм. Периодический вызов какой-либо функции в *1С* обеспечивается процедурой *ОбработкаОжидания.* Эта функция принимает в качестве первого аргумента имя процедуры, которая будет вызываться периодически, а в качестве второго – интервал вызова. Таким образом, *1С* позволяет вызывать в периоде только одну функцию. Для обеспечения вызова двух был разработан следующий алгоритм:

1) вычисляется НОД периодов выгрузки номенклатуры и загрузки заказов;
2) вычисленный НОД устанавливается в качестве интервала вызова;
3) в качестве функции вызова устанавливается функция проверки (*евпИтерацияСинхронизации*);
4) в *s*-методы *евпИтерацияСинхронизации* путем вычисления остатка деления временного отрезка, прошедшего с момента запуска синхронизации до настоящего момента, на период синхронизации производится проверка, какое действие должно быть выполнено на данной итерации (выгрузка номенклатуры или загрузка заказов); т. е. если с момента запуска синхронизации прошло 2 часа, а периоды равны 15 и 25 минутам, то вычисляются 120 *mod* 15 = 0, 120 *mod* 25 = 20; таким образом, на данной итерации вызывается выгрузка заказов.
5) при изменении параметров синхронизации производится перезапуск синхронизации;
6) синхронизация завершается при закрытии *1С*.

КОМПОНЕНТЫ dll

Набор компонентов *dll* представляет собой совокупность трех файлов:

1) *euripvk.dll* – внешняя компонента, содержит *d*-методы, написанные на языке *Delphi*;
2) *eurip.dll* – внутренняя компонента, содержит *c*-методы, написанные на языке *C/C++*;
3) *libmysql.dll* – компонента работы с СУБД.

Внешняя компонента. Внешняя компонента *EuripVK.dll* предназначена для взаимодействия *1С* с компонентой ядра. Она реализует функции, которые вызывают *c*-методы (функции ядра).

EuripVK.dll написан на языке *Delphi* и состоит из 4 модулей: *vk_object.pas, uBalloon.pas, addInLib.pas, addInObj.pas.*

В модуле *vk_object* реализованы функции, которые в качестве методов и свойств объекта *Euripteride* вызываются из *1С*. В классе *T_vk_object* реализованы методы *prop1, prop2,..* (для свойств) и *meth1, meth2, ...* (для методов). Все эти методы являются функциями, принимающими аргумент типа *TMode* (перечисление) и возвращающими строку.

Все *d*-методы *prop* и *meth* проверяют значение передаваемого параметра *mode* и в зависимости от этого выполняют определенные действия:

– если значение параметра *mode* равно *m_rus_name*, то возвращается русское название *s*-метода (или *s*-свойства);
– если значение параметра *mode* равно *m_en_name*, возвращается английское название;
– если значение параметра *mode* равно *m_n_params*, то в член класса *g_NParams* заносится число аргументов метода (имеет смысл только для *s*-методов);
– если значение параметра *mode* равно *m_get_value*, то выполняется *c*-метод (который должен вызываться на возвращение значения *s*-свойства) и в член класса *g_Value* заносится результат выполнения (имеет смысл только для *s*-свойств);
– если значение параметра *mode* равно *m_set_value*, то выполняется *c*-метод (который должен вызываться на установки значения *s*-свойства) (имеет смысл только для s-свойств);
– если значение параметра *mode* равно *m_execute*, то выполняется *c*-метод, который должен вызываться на вызов *s*-метода; необходимые аргументы возвращаются функциями:
– *GetParamAsInteger(<номер аргумента>)* для целых аргументов;
– *GetParamAsFloat(<номер аргумента>)* для вещественных аргументов;
– *GetParamAsString(<номер аргумента>)* для строковых аргументов.

Большинство *d*-методов *prop* предназначены только для возвращения информации в *1С*.

Все необходимые *c*-функции загружаются неявным образом (статически) из библиотеки *eurip.dll* (*external 'eurip.dll'*), а аргументы передаются с использованием порядка *cdecl*.

Файл addInObj. Файл *addInObj* предназначен для формирования объекта, загружаемого в *1С*. Он имеет следующие характеристики:

– в константе *c_PropCount* записано количество *d*-методов, соответствующих свойствам;
– тип *TProperties* представляет собой перечисление всех *d*-методов, соответствующих *s*-свойствам;
– в константе *c_MethCount* записано количество *d*-методов, соответствующих *s*-методам;
– тип *TMethods* предствляет собой перечисление всех *d*-методов, соответствующих *s*-методам;
– в методе *GetPropVal* класса *AddInObject* производится проверка аргумента *lPropNum* на соответствие типу и вызывается соответствующий ему *d*-метод-prop объекта *vk_object* с аргументом *m_get_value*;
– в методе *SetPropVal* класса *AddInObject* производится проверка аргумента *lPropNum* на соответствие типу и вызывается соответствующий ему *d*-метод-*prop* объекта *vk_object* с аргументом *m_set_value*;
– в методе *CallAsFunc* класса *AddInObject* производится проверка аргумента *lMethodNum* на соответствие типу и вызывается соответствующий ему *d*-метод-*meth* объекта *vk_object* с аргументом *m_execute*;
– в методе *GetNParams* класса *AddInObject* производится проверка аргумента *lMethodNum* на соответствие типу и вызывается соответствующий ему *d*-метод-*meth* объекта *vk_object* с аргументом *m_n_params*;

– в методе *FindProp* класса *AddInObject* производится проверка аргумента *bstrPropName*, для этого последовательно вызываются *prop*-методы объекта *vk_object*, результат сверяется с *bstrPropName*; когда необходимый *d*-метод находится, то его номер заносится в локальную переменную (затем возвращаемую по ссылке; неважно);

– в методе *FindMethod* производится проверка аргумента *bstrMethodName* (аналогично методу *FindProp* за исключением того, что вместо *prop*-методов используются *meth*) (табл. 2, 3).

Таблица 2

Методы *Prop* внешней компоненты

Английское название	Русское название	Назначение
counterpartId	*КодКонтрагента*	возвращает код текущего контрагента
counterpartName	*ИмяКонтрагента*	возвращает имя текущего контрагента
counterpartFullName	*ПолноеИмяКонтрагента*	возвращает полное имя текущего контрагента
counterpartType	*ТипКонтрагента*	Возвращает тип текущего контрагента (1 – юр.лицо, 2 – физ.лицо)
counterpartBankNumber	*БанковскийСчетКонтрагента*	возвращает банковский счет текущего контрагента
counterpartLegalPerson	*ЮрЛицоКонтрагента*	возвращает код юридического лица текущего контрагента
counterpartEmail	*ЭлАдресКонтрагента*	возвращает *e-mail* текущего контрагента
counterpartInn	*ИннКонтрагента*	возвращает ИНН текущего контрагента
counterpartOkpo	*ОкпоКонтрагента*	возвращает ОКПО текущего контрагента
counterpartFormAddress	*ЮрАдрессКонтрагента*	возвращает юридический адрес текущего контрагента
counterpartFactAddress	*ФактАдресКонтрагента*	возвращает фактический адрес текущего контрагента
counterpartTelephone	*ТелефонКонтрагента*	возвращает телефон текущего контрагента
counterpartComment	*КомментарийКонтрагента*	возвращает комментарий, соответствующий текущему контрагенту
counterpartLPC	*ЮлкКонтрагента*	возвращает комментарий, соответствующий юр. лицу текущего контрагента
maxCounterpartLegalPersonId	*МаксимальныйКодЮрЛица*	возвращает максимальный код контрагента
counterpartCountry	*ГосударствоКонтрагента*	возвращает государство контрагента
counterpartRegion	*РегионКонтрагента*	возвращает регион контрагента
counterpartCity	*ГородКонтрагента*	возвращает город контрагента
counterpartZip	*ПочтовыйИндексКонтрагента*	возвращает почтовый индекс контрагента
counterpartName1	*ВтороеИмяКонтрагента*	возвращает второе имя контрагента
newCounterparts	*НовыйКонтрагент*	производит проверку наличия следующей строки в списке контрагентов. Возвращает 1 – если есть, 0 – если нет
newBankNumbers	*НовыйБанковскийСчет*	производит проверку наличия следующей строки в списке банковских счетов. Возвращает 1 – если есть, 0 – если нет
bankNumberId	*КодБанковскогоСчета*	возвращает код текущего банковского счета
bankNumberName	*НаименованиеБанковскогоСчета*	возвращает название (тип) текущего банковского счета
bankNumberValue	*ЗначениеБанковскогоСчета*	возвращает значение текущего банковского счета (расчетный счет)
bankNumberC	*ХранилищеБанковскогоСчета*	возвращает банк, в котором открыт текущий счет
bankNumberOwner	*ВладелецБанковскогоСчета*	возвращает контрагента, которому принадлежит текущий банковский счет
orderId	*КодЗаказа*	возвращает код текущего заказа
orderCounterpart	*КонтрагентЗаказа*	возвращает код контрагента, сделавшего текущий заказ
orderNomenclature	*НоменклатураЗаказа*	возвращает номенклатуру, заказанную текущим заказом
orderCount	*КоличествоЭкземпляровЗаказа*	возвращает количество экземпляров номенклатуры, заказанной текущим заказом
orderPrice	*ЦенаЗаказа*	возвращает цену номенклатуры, заказанной текущим заказом
isNewOrder	*НовыйЗаказ*	производит проверку наличия следующей строки в списке заказов. Возвращает 1 – если есть, 0 – если нет
bankId	*КодБанка*	возвращает код текущего банка
bankName	*НаименованиеБанка*	возвращает название текущего банка
bankKNumber	*КорСчетБанка*	возвращает кор. счет текущего банка
newBanks	*НовыйБанк*	производит проверку наличия следующей строки в списке банков. Возвращает 1 – если есть, 0 – если нет
selfFirmId	*КодСвоейФирмы*	возвращает/устанавливает код собственной фирмы
valute	*КодВалюты*	возвращает/устанавливает код валюты
timerProcNumber	*НомерПроцедурыТаймера*	возвращает индекс процедуры, которая должна вызываться на текущий момент
timerPeriod	*ПериодТаймера*	возвращает период таймера

Таблица 3

D-методы, реализующие *s*-методы

Английское название	Русское название	Параметры	Назначение
init	*Инициализация*	Ip-адрес, порт, имя базы, имя пользователя, пароль, префикс таблиц, постфикс таблиц	подключение к СУБД
updateNomenclatureGroup	*ОбновлениеГруппы-Номенклатуры*	Код, имя, код родительской группы, комментарий	создание или изменение группы номенклатуры
updateNomenclatureGroupIm	*ОбновлениеГруппы-НоменклатурыИ*	Код, имя, код родительской группы, комментарий, файл с фотографией	расширенное создание или изменение группы номенклатуры
updateNomenclature	*ОбновлениеНоменклатуры*	Код, имя, код группы, комментарий, полное название, артикуль, остаток на складе	создание или изменение номенклатуры
updateNomenclatureIm	*ОбновлениеНоменклатурыИ*	Код, имя, код группы, комментарий, полное название, артикуль, остаток на складе, файл с фотографией, файл с большой фотографией	расширенное создание или изменение номенклатуры
updatePrice	*ОбновлениеЦены*	Номенклатура, тип, процент, валюта, значение	создание или изменение цены
updatePriceType	*ОбновлениеТипаЦены*	Код, название, вычисляемость (1 – если вычисляет, 0 – задается), базовый тип (для вычисляемых) процент, валюта, НДС, НП, комментарий	создание или изменение типа цены
updateCounterpart	*ОбновлениеКонтрагента*	Код, имя, тип (1 – юр.лицо, 2 – физ.лицо), код юр.лица (/физ.лица) основной банковский счет, e-mail, комментарий	создание или изменение контрагента
updateLegalPerson	*ОбновлениеЮрЛица*	Код, имя, ИНН\КПП, ОКПО, юр. адрес, фактический адрес, телефон, полное имя, комментарий	создание или изменение юридического лица
updateBankNumber	*ОбновлениеБанковскогоСчеа*	Код, название (тип) значение, банк, контрагент	создание или изменение банковского счета
readNewCounterparts	*ВыбратьНовыхКонтрагентов*		чтение из базы списка контрагентов
readNewBanks	*ВыбратьНовыеБанки*		чтение из базы списка банков
updateBank	*ОбновлениеБанка*	БИК название, кор. счет	создание или изменение банка
clearNomenclature	*ОчисткаНоменклатуры*		удаление всех групп и элементов номенклатуры
clearCounterparts	*ОчисткаКонтрагентов*		удаление всех контрагентов и юридических лиц
readNewBanksNumbers	*ВыбратьНовыеБанковскиеСчета*		чтение из базы списка банковских счетов
removeNomenclature	*УдалитьНоменклатуру*	код удаляемой номенклатуры	удаление элемента номенклатуры
removeNomenclatureGroup	*УдалитьГруппуНоменклатуры*	код удаляемой группы	удаление группы номенклатуры
removeCounterpart	*УдалитьКонтрагента*	код удаляемого контрагента	удаление контрагента
removeLegalPerson	*УдалитьЮрЛицо*	код удаляемого юр.лица	удаление юридического лица
removeBankNumber	*УдалитьБанковскийСчет*	код удаляемого счета	удаление банковского счета
removeBank	*УдалитьБанк*	код удаляемого банка	удаление банка
readNewOrders	*ВыбратьНовыеЗаказы*		чтение из базы списка заказов
startTimer	*ЗапускТаймера*		запуск таймера для обработки синхронизации
timerIteration	*НоваяИтерацияТаймера*		вызов новой итерации таймера

ЯДРО

Назначение и структура. Компонент *eurip.dll* предназначен для связи с СУБД, вызывая для этого функции библиотеки *libmysql.dll.* Схема его взаимодействия с другими компонентами представлена на рис. 4.

Компонент *eurip.dll* состоит из трех частей:

1) интерфейсного модуля, обеспечивающего импорт функций;

2) модуля *dbmanager* осуществляющего взаимодействие с СУБД;

3) библиотеки *scrvl*, использующейся для работы со строками.

Рис. 4. Схема взаимодействия компонентов

Интерфейсный модуль реализует функции, доступные для вызова из других компонент (*euripvk.dll*). Эти функции написаны на *C++* в режиме совместимости с *C*. Они вызывают методы глобального статического объекта *m_dbManager* – экземпляра класса *DBManager*, инкапсулирующего работу с СУБД (табл. 4).

Функции синхронизации определяют механизм вычисления периода, через который необходимо проводить синхронизацию. В качестве входных параметров здесь используются время ожидания (в секундах) перед выгрузкой номенклатуры, перед загрузкой заказов и периоды этих обработок. Алгоритм вычисления механизма синхронизации имеет неясный алгоритм (основанный на алгоритме Евклида), который по понятным причинам описан здесь не очень подробно. При необходимости вносить в него изменения (крайне не рекомендуется), следует изучать код реализации перечисленных здесь четырех методов.

Таблица 4

Интерфейсные методы библиотеки eurip.dll.

Название	Параметры	Назначение
1	2	3
initMySql	Ip-адрес, порт ,имя базы, имя пользователя, пароль, префикс, постфикс	подключение к базе
clearNomenclature		удаление всех записей о номенклатуре (и группах) из базы
clearCounterparts		удаление всех записей о контрагентах
updateNomenclatureGroup	Код, имя, код родительской группы, комментарий	создание или изменение группы номенклатуры
updateNomenclatureGroupImage	Код, имя, код родительской группы, комментарий, файл с фотографией	расширенное создание или изменение группы номенклатуры (происходит копирование фотографии)
updateNomenclature	Код, имя, код группы, комментарий, полное название, артикуль, остаток на складе	создание или изменение номенклатуры
updateNomenclatureImage	Код, имя, код группы, комментарий, полное название, артикль, остаток на складе, файл с фотографией, файл с большой фотографией	расширенное создание или изменение номенклатуры (происходит копирование фотографий)
updatePrice	Номенклатура, тип, процент, валюта, значение	создание или изменение цены
updatePriceType	Код, название, вычисляемость (1 – если вычисляет, 0 – задается), базовый тип (для вычисляемых), процент, валюта, НДС, НП, комментарий	создание или изменение типа цены
updateCounterpart	Код, имя, тип (1 – юр.лицо, 2 – физ.лицо), код юр.лица (физ.лица), основной банковский счет, e-mail, комментарий	создание или изменение контрагента
updateLegalPerson	Код, имя, ИНН\КПП, ОКПО, юр. адрес, фактический адрес, телефон, полное имя, комментарий	создание или изменение юридического лица
updateBankNumber	Код, название (тип) значение, банк, контрагент	создание или изменение банковского счета
updateBank	БИК, название, кор. счет	создание или изменение банка
readNewBanks		чтение списка банков из БД
selectBank	код банка	выбор конкретного банка из списка
isNewBanks		проверка на наличие банков за текущим
getBankId		возвращает код текущего банка
getBankName		возвращает название текущего банка
getBankKNumber		возвращает кор. счет текущего банка
readNewCounterparts		чтение из базы списка контрагентов
selectCounterpart	код контрагента	выбор конкретного контрагента из списка
isNewCounterparts		проверка на наличие контрагента за текущим
getCounterpartId		возвращение кода текущего контрагента

Окончание таблицы 4

1	2	3
getCounterpartTime		возвращение времени создания контрагента
getCounterpartName		возвращение имени текущего контрагента
getCounterpartType		возвращает тип текущего контрагента (1 – юр.лицо, 2 – физ.лицо)
getCounterpartName1		возвращение второго имени текущего контрагента
getCounterpartLegalPerson		возвращение кода юр.лица текущего контрагента
getCounterpartBank-Number		возвращение банковского счета текущего контр-агента
getCounterpartEmail		возвращение e-mail текущего контрагента
getCounterpartComment		возвращение комментария для текущего контр-агента
getCounterpartLPN		возвращение полного наименования юр.лица те-кущего контрагента
getCounterpartInn		возвращение ИИН текущего контрагента
getCounterpartOkpo		возвращение ОКПО текущего контрагента
getCounterpartCounrty		возвращение государства текущего контрагента
getCounterpartRegion		возвращение региона текущего контрагента
getCounterpartCity		возвращение города текущего контрагента
getCounterpartZip		возвращение почтового индекса текущего контр-агента
getCounterpartFormAddres s		возвращение юр. адреса текущего контрагента
getCounterpartFactAddress		возвращение фактического адреса текущего контрагента
getCounterpartFullName		возвращение полного имени текущего контрагента
getCounterpartLPC		возвращение комментария юр.лица имени текуще-го контрагента
readNewBankNumbers		чтение списка банковских счетов из БД
selectBankNumber	код банковского счета	выбор конкретного банковского счета
isNewBankNumbers		проверка на наличие банковских счетов за текущим
getBankNumberId		возвращение кода текущего банковского счета
getBankNumberName		возвращение названия (типа) текущего банковско-го счета
getBankNumberNumber		возвращение значения текущего банковского счета
getBankNumberBank		возвращение банка, в котором открыт текущий банковский счет
getBankNumberOwner		возвращение контрагента, которому принадлежит текущий банковский счет
readNewOrders		чтение списка заказов клиентов
isNewOrders		проверка на наличие заказов за текущим
getOrderId		возвращение кода текущего заказа
getOrderCounterpart		возвращение контрагента, сделавшего текущий заказ
getOrderNomenclature		возвращение номенклатуры текущего заказа
getOrderCount		возвращение количества элементов текущего заказа
getOrderPrice		возвращение цены текущего заказа
removeNomenclature	код номенклатуры	удаление номенклатуры
removeNomenclatureGroup	код группы номенклатуры	удаление группы номенклатуры
removeCounterpart	код контрагента	удаление контрагента
removeLegalPerson	код юр.лица	удаление юр.лица
removeBankNumber	код банковского счета	удаление банковского счета
removeBank	код банка	удаление банка
getSelfFirm		возвращение кода собственной фирмы
setSelfFirm	код фирмы	установка кода собственной фирмы
getValute		получение кода дефолтной валюты
setValute	код валюты	установка кода дефолтной валюты
getMaxCounterpartId		возвращение максимального кода контрагента
startTimer	ожидание перед первой выгрузкой но-менклатуры, ожидание перед первой загрузкой заказов, период между вы-грузками номенклатуры, период между загрузками заказов	получение периодов и времени ожидания. Вычис-ление по ним периода вызова процедуры синхро-низации
timerIteration		указание на следующую итерацию синхронизации
getTimersPeriod		возвращение текущего периода синхронизации
getTimersResult		возвращение результата работы таймера

DBManager. Обращения к СУБД производится при помощи вызова функций *mysql*. Для работы с этими функциями в методе *DBManager::pr_initFunction* происходит выгрузка библиотеки *libmysql.dll*, а затем получение адресов всех необходимых процедур и занесение их в члены класса. Для их определения описан ряд типов:

– *MYSQL * STDCALL (*tmysql_init) (MYSQL *mysql)*;
– *MYSQL * STDCALL (*tmysql_real_connect) (MYSQL *mysql, const char *host, const char *user, const char *passwd, const char *db, unsigned int port, const char *unix_socket, unsigned long clientflag)*;
– *int STDCALL (*tmysql_query)(MYSQL *mysql, const char *q)*;
– *void STDCALL (*tmysql_close)(MYSQL *sock)*;
– *MYSQL_RES * STDCALL (*tmysql_store_result) (MYSQL *mysql)*;
– *MYSQL_ROW STDCALL (*tmysql_fetch_row) (MYSQL_RES *result)*;
– *unsigned int STDCALL (*tmysql_num_rows) (MYSQL_RES *res)*;
– *unsigned int STDCALL (*tmysql_num_fields) (MYSQL_RES *res)*;

Соответственно среди полей класса определены их экземпляры:

– *tmysql_real_connect pmysql_real_connect*;
– *tmysql_query pmysql_query*;
– *tmysql_close pmysql_close*;
– *tmysql_init pmysql_init*;
– *tmysql_store_result pmysql_store_result*;
– *tmysql_num_rows pmysql_num_rows*;
– *tmysql_fetch_row pmysql_fetch_row*;
– *tmysql_num_fields pmysql_num_fields*;

Абстрагирование таблиц, производящееся в методе *DBManager::pr_createTablesNames*, позволяет использовать в качестве имен таблиц и полей переменные. Следует учитывать, что используемость полей определяется политикой интернет-магазина, поэтому в зависимости от версии реализации могут использоваться не все описанные поля. Кроме того, из-за экзотичности некоторых движков используемых интернет-магазинов могут добавляться дополнительные поля (табл. 5).

Работа с СУБД. Подключение к СУБД происходит при создании экземпляра *DBManager*. В качестве параметров конструктор этого класса принимает следующие поля: имя хоста (*ip*-адрес), порт, имя пользователя (логин), пароль, имя СУБД, префикс таблиц, постфикс таблиц.

Для удаления записей реализованы методы *clearNomenclature* и *clearCounterparts*, удаляющие все записи по номенклатуре и по контрагентам, соответственно. Кроме того, существуют методы, позволяющие удалять одну выбранную строку (по *id*): *removeNomenclature*, *removeNomenclatureGroup*, *removeCounterpart*, *removeLegalPerson*, *removeBankNumber*, *removeBannk*. Все они вызывают защищенный метод (*pr_removeElement*).

Для внесения изменений реализован ряд методов (их параметры соответствуют параметрам методов из табл. 4 и потому не приведены здесь) *updateNomenclatureGroup*, *updateNomenclature*, *updatePrice*, *updatePriceType*, *updateCounterpart*, *updateLegalPerson*, *updateBankNumber*, *updateBank*, *setSelfFirm*, *setValute*. Работа этих методов реализована следующим образом:

1) происходит проверка на наличие в таблице строки с соответствующим *id*, для этого применяется метод *pr_isThere;*

2) если соответствующая запись не найдена, данные добавляются командой *INSERT*, если же строка с таким *id* существует, то происходит обновление (*UPDATE*);

3) *DBManager* предоставляет несколько методов для получения одиночных данных из СУБД:

– *getSelfFirm* – получение кода собственной фирмы;
– *getValute* – получение кода дефолтной валюты;
– *getMaxId(CrVString)* – получение максимального *id* в данной таблице.

Таблица 5

Поля таблиц СУБД в DBManager

Название поля	Значение
nomenclatureGroup	группы номенклатуры
id	код
name	название
parent	код родительской группы
comment	комментарий
products_count	количество номенклатур в данной группе
image	фотография группы
nomenclature	номенклатуры
id	код
name	название
fullname	полное наименование (для печати)
parent	код группы
comment	комментарий
price	цена
image	фотография
image1	большая фотография
rating	рейтинг (0–5)
votes	количество голосов, определяющих рейтинг
enabled	видимость товара (1 – виден, 0 – нет)
articul	артикул
rem	остаток товара
counterpart	контрагент
id	код
name	имя
othername	другое имя
email	e-mail
country	государство
region	регион
city	город
address	адрес
telephone	телефон
zip	почтовый индекс
time	время регистрации
orders	заказы
id	код
nomenclature	код номенклатуры
count	количество экземпляров
price	цена
counterpart	контрагент

Выгрузка элементов (списков) из СУБД производится следующим способом: вначале происходит извлечение данных и сохранение их члене-класса *m_result,* а затем интерфейсные функции обращаются к этим данным и возвращают их элементы.

Список методов, извлекающих данные (данные методы вызываются из соответствующих интерфейсных методов *read* (*readNewBanks*, *readNewCounterparts*, *readNewOrders*). Прочие интерфейсные методы занимаются исключительно обработкой массива *m_result.*: *firstCounterpart*, *firstBank*, *firstOrder*, *firstCounterpartOrder.*

Для вызова данных из СУБД реализован метод *pr_firstObject*, вызываемый всеми вышеперечисленными методами. При выборе строки она удаляется из базы. Член-класса *m_result* представляет собой двумерный массив (строки данных и поля).

LibmySQL. Библиотека *libmySQL.dll* реализует функции, позволяющие взаимодействовать с СУБД. Как и всякая скомпилированная библиотека, *libmySQL* содержит только реализации, а не описания и типы, которые содержатся в заголовочном файле *mysql.h.* При разработке ядра этот файл (и несколько других) был взят из *MySQLServer* и помещен в директорию с заголовочными файлами среды.

ВЫВОДЫ

Таким образом, разработана технология интеграции системы управления предприятием *1С* и интернет-магазинов, базирующихся на различных движках.

ЛИТЕРАТУРА

1. *Рязанцева Н., Рязанцев Д.* 1С Предприятие: Торговля и склад. Секреты работы. СПб.: БХВ-Петербург, 2003. 368 с.

Поступила в редакцию 25 марта 2014 г.

Kryuchin O.V., Khabirova K.R., Vyazovova E.V. INTEGRATION OF INTERNET-STORE AND ENTERPRISE MANAGEMENT SYSTEM WITH MODULES *1C* EURIPTERIDE

The technology of enterprise management system *1C* and internet-stores based on different equipment is described. The full description of the system architecture and its technical characteristic specificity is given.

Key words: internet-stores; management system; messages passing protocol.

Крючин Олег Владимирович, Тамбовский государственный университет им. Г.Р. Державина, г. Тамбов, Российская Федерация, магистрант по направлению подготовки «Прикладная математика и информатика» института математики, физики и информатики, e-mail: kryuchov@gmail.com

Kryuchin Oleg Vladimirovich, Tambov State University named after G.R. Derzhavin, Tambov, Russian Federation, Candidate for Master's Degree of Direction of Preparation of "Applied Mathematics and Informatics" of Mathematics, Physics and Informatics Institute, e-mail: kryuchov@gmail.com

Хабирова Кристина Раильевна, Тамбовский государственный университет им. Г.Р. Державина, г. Тамбов, Российская Федерация, студентка института математики, физики и информатики, e-mail: kryuchov@gmail.com

Khabirova Kristina Railyevna, Tambov State University named after G.R. Derzhavin, Tambov, Russian Federation, Studewnt of Mathematics, Physics and Informatics Institute, e-mail: kryuchov@gmail.com

Вязовова Елена Владимировна, Тамбовский государственный университет им. Г.Р. Державина, г. Тамбов, Российская Федерация, магистрант по направлению подготовки «Прикладная математика и информатика» института математики, физики и информатики, e-mail: kafedra_kmm@mail.ru

Vyazovova Elena Vladimirovna, Tambov State University named after G.R. Derzhavin, Tambov, Russian Federation, Candidate for Master's Degree of Specialty of "Applied Mathematics and Informatics" of Mathematics, Physics and Informatics Institute, e-mail: kafedra_kmm@mail.ru