Обзор функциональных возможностей современных систем, позволяющих создавать и использовать электронные формы

В. А. Скорняков

В статье рассматриваются современные системы, позволяющие создавать и использовать электронные формы и рассчитанные на конечного пользователя. Рассматриваются возможности средств, предоставляемых этими системами для создания и редактирования электронных форм, а также возможности самих создаваемых форм. Производится анализ систем, а затем — их сравнительный анализ по основным характеристикам электронных форм и средств создания этих форм с помощью ввода критериев сравнения рассматриваемых характеристик.

Наиболее гибкими системами, позволяющими создавать и использовать электронные формы, являются системы разработки ПО (программного обеспечения), такие как Microsoft Visual Studio или Microsoft Visual Basic. Но подобные системы требуют значительной квалификации разработчика, а также при любом изменении необходима перекомпиляция приложений. К тому же они не рассчитаны на конечного пользователя, поэтому системы разработки ПО в данном обзоре не представлены.

Рассмотрим следующие аспекты использования электронных форм: внешний вид форм и удобство их заполнения, способы и удобство редактирования форм, форматы описания форм и структурированность данных, вводимых с помощью форм.

Microsoft Word и VBA

Microsoft Word — мощный текстовый процессор, ориентированный на создание и редактирование документов для последующего вывода на печать. Поэтому документы, подготавливаемые с помощью этого редактора, имеют такой же вид, какой будет при их выводе на печать. В связи с этим, формы, которые позволяет создавать и заполнять Microsoft Word, ограничены двумерным пространством. Кроме того, поля для ввода переменных данных в

документах Microsoft Word, имеют точно такое же расположение, как и сами данные в итоговом текстовом документе. Подобные формы будем называть *бумажными*. Бумажные формы удобны тем, что пользователь при заполнении всегда видит конечный текстовый документ. Минус такого подхода заключается в том, что бумажные формы ориентированы на представление конечного результата, а не на удобство заполнения таких форм, т. к. бумажная форма создается в соответствии со структурой текстового документа.

Для создания формы в Microsoft Word необходимо сначала создать шаблон документа [5]. После этого, в созданном шаблоне размещаются статические элементы формы и поля для ввода текстовых переменных данных. Набор полей для ввода текстовых данных сильно ограничен тем, что вводимые данные должны отображаться так, как они будут выглядеть в результирующем текстовом документе. Для форм Microsoft Word имеется возможность создания макросов на языке программирования VBA, с помощью которых можно наладить взаимосвязи полей, проверку вводимых данных и использование дополнительных инструментов заполнения полей формы. Формат шаблонов документов Word является закрытым.

Кроме форм, внедренных в шаблоны документов, Microsoft Word позволяет создавать макросы на VBA, которые могут выполняться при возникновении какого-либо события в среде текстового процессора. Макросы могут использовать электронные формы для взаимодействия с пользователем. С помощью макросов и электронных форм можно писать сложные приложения для Microsoft Word. Данные каждой конкретной формы никак не оформлены и содержатся частями в каждом отдельном элементе формы только в течение жизни самой формы. Поэтому каждое конкретное приложение на VBA для Word может правильно интерпретировать данные, вводимые в форму.

Формы для макросов Word имеют вид диалога ОС Microsoft Windows и редактируются с помощью визуального редактора экранных форм, схожего с редактором форм системы программирования Microsoft Visual Basic. Эти формы могут содержать любые элементы управления Active-X, зарегистрированные в системе. Следовательно, формы не ограничены двумерным пространством и могут содержать элементы управления любой сложности и внешнего вида.

Экранные формы для макросов Microsoft Word состоят из двух файлов с расширениями frm и frx. Файл frm является текстовым и содержит основные параметры самой формы: имя, размер и пр. Файл frx является бинарным и содержит описание состава и свойств всех элементов управления формы.

Microsoft InfoPath

Приложение MS InfoPath является частью пакета приложений для офиса Microsoft Office [1], однако его правильно рассматривать как часть комплексной информационной системы, решающую задачу ввода информации и частично вывода. Фактически, InfoPath является визуальным инструментом конструирования и заполнения форм [12].

InfoPath позволяет создавать довольно гибкие формы-документы, которые могут быть распечатаны или переданы по сети. Форма создается с помощью визуального редактора, схожего с редактором Visual basic. В форму документа InfoPath включены макеты и элементы управления. Макеты позволяют разметить документ в виде таблиц, в каждой ячейке которых может находиться своя простая статическая форма. Макеты позволяют создавать формы, в которых одни отдельные области могут являться необязательными и свернутыми, а другие — повторяться многократно при заполнении формы данными. По нарисованной форме можно настроить источник данных или можно сначала настроить источник данных, а затем по нему нарисовать форму. Для каждой формы имеется возможность написать скрипт на языках JScript или VBScript для обработки событий, связанных с формой и ее элементами. В заключение можно создать несколько представлений данных, вводимых с помощью созданной формы. С помощью представлений можно нарисовать одну форму удобную для заполнения, а другую — для печати данных. Недостатком форм InfoPath является их двумерность и ограниченный набор элементов управления.

Каждое из представлений формы InfoPath хранится в виде отдельного XSLT-файла. InfoPath позволяет экспортировать все необходимые данные документа в набор XML-файлов определенного формата.

HTML и Web-формы

HTML является стандартом представления информации в Web [7]. Начиная с версии 2.0 в HTML появилась поддержка интерактивных форм, позволяющих организовывать сбор информации через Web. Форма является частью HTML документа. Web-форма может содержать некоторый стандартный набор элементов управления, Java-апплеты, позволяющие реализовать другие элементы управления и любые элементы управления Active-X. Web-форма может работать под управлением JavaScript, с помощью которого возможно реализовать проверку вводимых данных или динамически изменять форму.

HTML документы имеют текстовый формат и интерпретируются и отображаются на экране Интернет-браузерами, такими как Microsoft Internet Explorer или Netscape Navigator. Интернет-браузеры разбирают текст документа на HTML и создают на экране некоторое представление этого документа в виде текста и доступных элементов управления. Некоторые браузеры позволяют выводить HTML-документы на экраны карманных компьютеров или на экран телевизора. В результате внешний вид документа сильно зависит от размеров экрана браузера и от средств системы, в которой браузер работает. Получаемые представления документов HTML регламентируются консорциумом W3C. Web-формы предназначены для сбора данных и отправки их на сервер и обычно не приспособлены для вывода на печать.

Создание Web-форм возможно вручную в простейшем текстовом редакторе или с помощью современных средств создания HTML-документов, таких, как Microsoft FrontPage или Macromedia Dreamweaver. Редакторы HTML-документов позволяют разметить документ с помощью таблиц и стилей [4] и разместить в нем все необходимые элементы Web-формы с помощью визуального редактора. После создания внешнего вида есть возможность воспользоваться редакторами и отладчиками скриптов для создаваемых документов, а также текстовым редактором для ручной правки полученных документов.

Lotus Notes

Клиент-серверные системы IBM Lotus Notes (Notes, Lotus Domino Web Access ™ и Lotus Domino Access for Microsoft Outlook) предназначены для тех пользователей, которым требуется доступ к информации, соответствующий их роли, кругу обязанностей или стилю работы. Система IBM Lotus Notes — флагман систем Lotus, который произвел настоящую революцию в методах работы и сотрудничества людей. Lotus Notes и Lotus Domino предлагают надежные инструменты для организации коллективной работы и обмена сообщениями, объединив средства электронной почты, ведения календарей, группового планирования, управления контактами и заданиями, Web-браузера — причем все это в рамках настраиваемой, удобной в использовании среды [9], [10].

Электронная форма документа является одним из центральных понятий Lotus Notes. Каждая форма в базе данных Notes представляет собой бланк для ввода, чтения, редактирования и печати документов определенного типа. Форма сама может содержать множество различных элементов

(объектов), таких как поля, статический текст, графические изображения, OLE-объекты, кнопки и секции. В 4-й версии Notes к этому списку добавляются макеты, свертываемые секции и субформы. БД может иметь множество форм, используемых с различными типами документов или для представления одних и тех же документов в разных форматах [8].

Электронная форма документа Lotus Notes имеет важное значение в этой системе, т. к. форма содержит в себе все признаки, отвечающие за [6]:

- политику безопасности документов (уровень доступа различных пользователей к документу, создаваемому по этой форме, ключи шифрования и пр.);
- политику совместной работы с документами;
- тип окна, в котором будет выводиться форма;
- что делать с документом при закрытии формы (отправить документ почтой и сохранить в БД, отправить почтой, но не сохранять, сохранить, не отправляя, не отправлять и не сохранять изменения);

  и другие важные атрибуты документов.

Форма в Lotus Notes может содержать поля ввода, текст, таблицы, графические изображения, OLE-объекты, кнопки, активные участки, субформы. Наиболее важным элементом формы Lotus Notes является поле ввода, которое является частью текста документа. Для поля ввода можно указать следующие свойства:

- один из восьми определенных типов данных;
- формат отображения чисел;
- если поле имеет тип Rich Text, то оно может содержать стилизованный текст, таблицы, графику, гипертекстовые ссылки, активные участки, кнопки, свертываемые секции, OLE-объекты;
- список возможных значений (словарь), при этом для создания списка можно использовать простые списки, формулы, синонимы (да|1), список ключевых слов, составленный из всех имен групп и ролей, представленных в списке управления доступом текущей базы данных;
- формулу для заполнения поля;
- формулу для проверки данных, вводимых в поле;
- подсказку для поля;
- признаки шифрования поля и признак участия поля в шифровании всего документа.

Поле ввода на форме неброско выделяется среди текста формы двумя уголками, расположенными как русские прописные кавычки. При вводе текста в поле, границы поля перемещаются в тексте как обыкновенные символы. Некоторые поля можно сделать общими, т. е. в форме будет указана ссылка на поле, а его описание будет хранится в отдельном месте БД.

Интересной особенностью форм является наличие субформ-форм. То есть внутри каждой формы можно использовать созданную ранее форму и хранящуюся отдельно. При этом субформа может быть не постоянна, а вычисляться в зависимости от контекста использования формы.

На форме Lotus Notes можно задать элементы макета — ограниченные области для группы полей и абзацы, которые могут скрываться в зависимости от режима использования формы (редактирование, чтение, печать, передача по электронной почте).

Для гибкости проектирования форм Lotus Notes, существует набор событий, связанных с формой или ее элементами, для которых можно написать скрипт на языке LotusScript.

Форма создается в удобном визуальном редакторе с использованием различных панелей и диалогов свойств. Данные документов, создаваемых с помощью форм Lotus Notes имеют текстовый формат и не имеют описанной структуры.

Adobe Acrobat и PDF

Формат PDF (Portable Document Format) [11] — это формат сохранения электронных документов, разработанный компанией Adobe, который позволяет описывать представление документа независимо от платформы или устройства отображения документа. PDF-документ состоит из коллекции объектов, которые все вместе описывают внешний вид одной или нескольких страниц, возможно содержащих интерактивные элементы и дополнительную информацию для приложений. Страницы документа могут содержать в любых комбинациях текст, графику и изображения. В дополнение к статическим описаниям, документ может содержать интерактивные элементы, которые доступны только в электронном представлении документа.

Начиная с версии 1.2, в формате PDF поддерживаются интерактивные формы — коллекции полей, предназначенных для интерактивного получения информации от пользователя. PDF-документ может содержать любое количество полей, появляющихся на любой комбинации страниц документа. Произвольный набор этих полей может быть импортирован в

документ и экспортирован из него. Поля могут быть организованы в иерархию и могут наследовать атрибуты своих родителей в иерархии полей. Подчиненные поля в иерархии могут включать в себя дополнительные указания, которые определяют их вид на странице.

Интерактивные формы PDF могут содержать поля одного из следующих типов:

- Кнопки, выключатели и переключатели.
- Текстовые поля. Области, в которые пользователь может вводить текст.
- Поля выбора. Содержат несколько текстовых значений, одно из которых может быть выбрано в качестве значения. Например, списки или комбинированные списки.
- Поля подписи. Используются для создания электронных подписей, обеспечивающих идентификацию пользователей.

Использование интерактивных форм в PDF-документах обеспечивает структурирование информации в сочетании с точным описанием их внешнего вида и контролем внутренней целостности документа. Особенностью формата PDF является то, что документы в этом формате не могут быть модифицированы стандартными средствами, за исключением изменения значений в полях формы. Использование подпрограмм на Java Script, включенных в тело документа, позволяет гибко описывать зависимости между полями формы, контролировать внутреннюю целостность документа.

НИКА-технология

Система НИКА-технология, разработанная в ИСА РАН [18] (торговая марка XNika), предоставляет широкие возможности по настройке систем, обеспечивающих ввод и обработку сложноструктурированных документов. Документы в XNika имеют иерархическую структуру, которая может изменяться в процессе существования системы. Для заполнения и отображения документов используются экранные формы. Экранные формы внешне схожи с диалоговыми окнами ОС Windows. Формы XNika позволяют вводить и просматривать данные документов в удобном для пользователя виде и не зависят от печатного представления этих данных. Данные документов системы хранятся в БД НИКА, имеющей иерархическую структуру и передаются виде текста в формате XML.

Для настройки схемы БД и форм системы, построенной на основе XNika используется дизайнер сложноструктурированных форм. Дизайнер позволяет создавать модели содержания документов системы и формы, отображающие эти документы. Для создания модели содержания и форм в дизайнере имеются соответствующие визуальные редакторы, с помощью которых можно легко и быстро настроить схему данных и внешний вид документов. Для создания схемы данных и форм не обязательно полностью создавать схему и формы. Зачастую достаточно создать схему данных, после чего дизайнер позволит автоматически получить форму для отображения этих данных. Или наоборот — возможно сначала нарисовать форму, а затем получить схему данных, которую сможет отображать созданная форма.

При создании форм документов дизайнер позволяет использовать некоторый набор стандартных элементов ввода данных, а также любые зарегистрированные в ОС элементы управления Active-X. Кроме элементов ввода данных, на форме можно разместить элементы управления формой, такие, как набор закладок, вложенные списковые, динамические и статические формы. Вложенные формы могут использовать формы, созданные ранее и хранящиеся в хранилище форм. Для некоторых элементов ввода возможно настроить словари возможных значений.

Поля ввода (как отдельные, так и в составе таблиц) имеют широкий набор свойств:

- Один из 18 поддерживаемых и расширяемых типов данных;
- Способ редактирования (клавиатура или какой-либо инструмент ввода данных в виде элемента управления Active-X);
- Словарь возможных значений;
- Подсказка для поля;
- Формулы для вычисления начального поля, пересчета значения в процессе заполнения формы, необходимости заполнения поля и проверки значения на корректность;
- Маска для вводимого текста;
- Набор общепринятых свойств, таких, как цвет, шрифт, выравнивание;
- Неограниченный набор пользовательских свойств, список которых можно задать в специальном окне дизайнера.

Для настройки словарей формы используется отдельное окно, которое позволяет настроить состав словарей, используемых полями формы, и

свойства словарей в контексте формы. При указании синонимов в словарях (Да|1; Нет|0) можно настроить форму таким образом, чтобы при выборе значения из словаря, синонимами заполнялось несколько соседних полей формы или поле принимало значение одного из синонимов выбранного в словаре слова. Словари могут храниться в виде файлов, а могут использовать пользовательские библиотеки, реализующие объект, поддерживающий необходимый COM-интерфейс. Недавно появилась возможность подключать с помощью дизайнера компоненты, которые не только наполняют словари значениями, но также еще и отображают собственное окно для выбора значений, что дает возможность создания и использования иерархических словарей.

Для наиболее гибкой настройки работы формы дизайнер предоставляет возможность создания и использования скриптов на языках VBScript или JavaScript. Скрипты позволяют оперировать как с элементами управления формы, так и с данными формы.

Формы документов системы XNika хранятся в специальном хранилище в виде файлов, имеющих формат XML и COM-хранилища. Статические словари хранятся в виде текстовых файлов.

Oracle Forms

Oracle Forms — это программный продукт корпорации Oracle, предназначенный для создания приложений баз данных, в котором создаются модули форм. Помимо этого, он включает в себя оболочку для разработки меню и библиотечных модулей PL/SQL [13].

Формы, создаваемые с помощью Oracle Forms могут иметь следующие элементы:

- Модуль формы — основной компонент интерактивных приложений. С точки зрения внутренней структуры, это — самый сложный модуль, состоящий из отдельных объектов различных видов (триггеры, блоки, окна и т. д.), которые мы кратко рассмотрим ниже. Форма представляет собой приложение (файл с расширением .fmx), которое может выполняться в среде Oracle, либо приложение, сгенерированное на языке Java, которое может выполняться в среде интерпретирования этого языка. Модуль формы может содержать:
  - — Элементы — первичные единицы для построения форм;
  - — Триггеры — блоки PL/SQL, связанные с другими объектами: с формой, с блоком данных или с элементом блока данных. Триггер

активизируется (fires), или выполняется, при наступлении конкретного события;

— Блоки данных — промежуточные единицы для построения форм. Блок данных можно рассматривать в двух аспектах: как совокупность элементов и как совокупность записей, имеющих одинаковую структуру. Для блока указывается количество одновременно отображаемых записей, а также способ их вывода (горизонтально или вертикально);

— Отношения — специальные объекты, которые используются в Forms для структурирования форм типа «основа — деталь». Эти объекты принадлежат основному блоку данных и отражают взаимосвязи, существующие между основной и соответствующими подчиненными записями. Главными свойствами отношения являются имя подчиненного блока данных и условие соединения, используемое в Forms для управления отношением;

— Объекты-основы и окна — фон, на котором размещаются шаблонный текст и элементы. В таблице свойств элемента указывается только одна канва. Элементы блока данных можно распределять среди различных объектов-основ;

— Объекты программирования — ряд объектов, используемых для структурирования программ. Такие объекты применяются как в модулях форм, так и в модулях меню, а некоторые из них — присоединенные библиотеки (attached libraries) и программные единицы (program units) — используются еще и в библиотечных модулях;

- Модуль меню — представляет собой упорядоченный набор элементов, которые аналогичны элементам меню любой оконной среды разработки — это разделители, подменю, выбор вариантов, триггерные элементы и т. д.

Элементы формы могут быть следующих типов:

- ActiveX Control (управляющий элемент ActiveX) — Отображает управляющий элемент ActiveX.
- Display Item (элемент отображения) — Текстовые элементы «только для чтения», отображающие данные и запрещающие пользователям вносить изменения.
- Image (изображение, образ) — Показывает растровое или векторное изображение.

- List Item (списочный элемент) — Список строк текста, которые можно отобразить как очередь (всплывающий список), как список фиксированного размера или как поле со списком.
- OLE Container (контейнер OLE) — Отображает объект OLE.
- Push Button (командная кнопка) — Дает пользователю возможность выполнять триггер, связанный с данной кнопкой (When-Button-Pressed).
- Radio Group (группа переключателей) — Дает пользователю возможность включать одну кнопку из группы, отключая при этом все остальные; данному объекту принадлежит набор элементов, называемых переключателями (radio buttons) и работающими в совокупности; каждый из этих элементов представляет конкретное значение группы; группа, в свою очередь, представляет столбец базы данных.
- Sound (звук) — Отображает значок, представляющий звук; щелчок мыши на этом значке вызывает звуковое сопровождение.
- Text Item (текстовый элемент) — Отображает текст в одно или многострочном поле индикации.
- User Area (область пользователя) — Отображает все, что пользователь может разместить в этом пространстве с помощью программы.

Элементы формы могут оперировать с данными следующих типов:

- Alpha — Любой буквенный символ верхнего или нижнего регистра.
- Char — Любой буквенный или числовой символ (соответствует типу CHAR1 в Oracle).
- Date — Правильные дата и время (соответствует типу DATE в Oracle).
- Int — Любое целое значение со знаком или без знака.
- Long — Строка символов увеличенной длины.
- Number — Число с фиксированной или плавающей точкой, со знаком или без знака, в экспоненциальном или в обычном представлении (соответствует типу NUMBER в Oracle).

С помощью Oracle Forms можно создать небольшое приложение, которое позволит пользователю просматривать или изменять данные БД Oracle. Все данные приложения хранятся в БД, имеющей известную заранее структуру, поэтому формы создаются с использованием структуры уже готовых таблиц БД.

Характерным элементом Oracle Forms является блок данных. Блоки данных бывают двух видов [13]. *Блок данных базовой таблицы* (*base-table data block*) соответствует таблице или представлению базы данных и задает некоторое число записей — строк этой таблицы или этого представления. *Управляющий блок* (*control block*) не соответствует таблице или представлению, и его записи не соответствуют строкам базы данных. Управляющие блоки представляют собой, как правило, совокупность элементов, каждый из которых состоит из одного значения. Следовательно, для управляющего блока необходима лишь одна строка. Например, если для группы записей нужно отслеживать некоторое составное (скажем, итоговое или среднее) значение, то в управляющем блоке создается элемент, представляющий это значение, причем для данного элемента данного управляющего блока определено только одно это значение.

Основной функцией блоков данных базовой таблицы является установление связи с таблицей или с хранимой процедурой базы данных. В Developer/2000 предусмотрена программа-мастер (wizard) для блоков данных, которая позволяет строить эти блоки из схемы базы данных. Кроме того, мастер помогает создавать блоки данных типа «основа — деталь». Developer/2000 автоматически управляет информацией базы данных, конструируя SQL-операторы и беря при этом за основу блоки данных и их структуру.

С помощью блоков данных можно строить так называемые запросы по примеру (query-by-example). В этом случае пользователь устанавливает для блока режим Enter Query, а затем вводит критерии запросов в поля специальной записи-примера. Затем при выполнении запроса Forms создает из этих значений SQL-оператор SELECT. В комбинации с конструкциями WHERE и ORDER BY это предоставляет широкие возможности по конструированию запросов. Кроме того, в режиме Enter Query пользователь может вводить собственные дополнительные SQL-команды в условие и указывать количество записей для буфера памяти, а также максимальное время выполнения запроса и максимальное количество записей, считываемых в форму.

Oracle Forms позволяет создавать формы с помощью визуального редактора форм. После редактирования формы необходимо сгенерировать исполняемый FMX-файл. Это связано с тем, что Oracle генерирует приложения в псевдокоде (файлы с расширением FMX), запуск которых возможен посредством Forms Runtime — небольшого пакета, устанавливаемого на клиентскую машину [14]. Сгенерированный FMX-файл можно поместить в любой каталог сервера приложений.

Сравнительный анализ

Для сравнения возможностей форм перечисленных выше систем будем использовать следующие критерии:

- Полнота реализации возможностей элементов управления;
- Структурированность данных, обрабатываемых с помощью форм;
- Полнота реализации формул для полей форм;
- Возможности программирования форм;
- Автоматизация создания форм на основе структуры данных и обратно;
- Гибкость настройки и наполнения используемых словарей;
- Типизирование обрабатываемых данных;
- Использование пользовательских свойств элементов формы.

Рассмотрим подробно перечисленные критерии сравнения систем.

Полнота реализации возможностей элементов управления

Под полнотой реализации возможностей элементов управления будем понимать меру возможности использования тех или иных элементов управления, предоставляемых ОС Windows или созданных с помощью каких-либо средств создания элементов управления. За меру возможности использования элементов управления возьмем процент использования следующих элементов:

1. Поля ввода текста и поля со статическим текстом;
2. Кнопки, флажки (выключатели);
3. Переключатели (радиокнопки);
4. Простые списки (списки текстовых значений);
5. Таблицы;
6. Закладки;
7. Списки подформ;
8. Уже готовые формы;
9. Скрываемые или открываемые по требованию области форм;
10. Словари;
11. Элементы управления Active-X;

12. OLE-объекты;
13. Апплеты;
14. Звуковое сопровождение;
15. Рисунки;
16. Всплывающие подсказки к полям;
17. Элементы прокрутки (slider, spin);
18. Рамки;
19. Динамическе поля (изменяющие размеры во время ввода данных).

Возможности использования перечисленных элементов указаны в табл. 1.

Структурирование вводимых данных

В различных системах набор данных, водимых или отображаемых с помощью форм, формируется каким-то определенным образом. Иногда этот набор представляется в виде простого списка значений, а иногда — в виде сложных иерархических структур. Рассматривая правила формирования наборов данных для форм, можно выделить несколько их разновидностей:

- Неопределенный (данные хранятся в элементах формы, а их последующая обработка возлагается на разработчика системы);
- Неструктурированный (неделимый поток данных);
- Одноуровневый список значений (набор пар Имя — Значение);
- Таблица;
- Иерархическая структура.

Рассматриваемые в статье системы предназначены для решения различных задач. Поэтому данные, получаемые из форм, формируются способом, наиболее целесообразным для конкретных задач. Для сравнения систем интересна степень структурированности данных. Введем степень структурированности данных для форм. Для определения понятия структурированности данных, я ввел меру структурированности в соответствии с определением структуры данных.

В результате, для меры структурированности я использовал 5 значений: 1 — Неструктурированный; 2 — Одноуровневый список; 3 — Таблица; 4 — Иерархическая структура. Значения этой меры для различных систем приведены в табл. 2.

Таблица 1

Полнота реализации возможностей элементов управления

Элемент / Система	Текстовые поля	Кнопки, флажки	Переключатели	Простые списки	Таблицы	Закладки	Списки подформ	Уже готовые формы	Скрываемые по требованию области форм	Словари	Элементы управления Active-X	OLE-объекты	Апплеты	Звуковое сопровождение	Рисунки	Всплывающие подсказки к полям	Элементы прокрутки	Рамки	Динамические поля	полнота реализации возможностей (%)
Microsoft Word	+	–	–	–	+	–	–	–	–	–	+	+	–	–	+	–	–	+	–	31,6
Microsoft VBA и диалоговые формы	+	+	+	+	+	+	–	+	–	+	+	+	–	–	+	–	+	+	–	68,4
Microsoft InfoPath	+	+	+	+	+	–	+	–	+	+	+	+	–	–	+	+	–	+	+	73,7
Web-формы	+	+	+	+	+	–	–	–	+	+	+	–	+	+	+	+	–	+	–	68,4
Lotus Notes	+	+	+	+	+	–	–	+	–	+	–	+	–	–	+	+	–	+	–	57,9
Adobe Acrobat	+	+	+	+	–	–	–	–	–	+	–	–	–	–	+	–	–	+	–	36,8
Oracle Forms	+	+	+	+	–	–	–	–	–	+	+	+	–	+	+	–	–	+	–	52,6
НИКА-технология	+	+	–	+	+	+	+	+	+	+	+	–	–	–	+	+	–	+	+	73,7

Полнота реализации формул для полей форм

Для удобства редактирования форм некоторые рассматриваемые системы предоставляют возможность формульного описания значений некоторых параметров. Формульное описание значительно нагляднее и проще использования языков программирования, т. к. формула состоит из одной строки и является математическим выражением, в которое в качестве параметров входят имена полей. Рассматриваемые системы позволяют указывать формулы для следующих атрибутов элементов формы:

- Значение;
- Значение по умолчанию;
- Корректность введенного значения;
- Обязательность заполнения формы;
- Релевантность значения элемента (имеет ли смысл это значение);
- Произвольный атрибут.

Кроме наличия атрибутов, значение которых можно задать с помощью формулы, также можно рассмотреть полноту и расширяемость набора функций, поддерживаемых формулами. Для этого введем оценку полноты набора функций для формул по пятибалльной шкале:

- Неудовлетворительно — невозможно использовать функции в формулах;
- Удовлетворительно — присутствует минимальный набор функций (Avg, Sum, Min, Max, Count);
- Хорошо — присутствует расширенный набор функций (Avg, Sum, Min, Max, функции работы с датами, функции ветвления (аналог iif в Бейсике));
- Отлично — присутствует расширенный набор функций и возможность расширения набора пользовательскими функциями.

Для того, чтобы сравнить полноту реализации формул для элементов форм в различных системах, введем меру полноты реализации формул в виде пар вида: (оценка полноты набора функций для формул; количество атрибутов, для которых можно задать формулу). Значения этой меры для различных систем приведены в табл. 2.

Реализация возможности использования скриптов для форм

Почти все рассматриваемые системы позволяют программировать поведение форм с помощью программ на скриптовых языках — VBScript или JavaScript. Исключение составляет система Oracle Forms, в которой используется язык PL\SQL.

Автоматизация создания форм на основе структуры данных и создания структуры данных по созданной экранной форме

Для систем, не имеющих структурированности данных форм, этот раздел не имеет смысла. Для систем же, которые используют некоторую схему данных, отдельную от форм, автоматизация создания данных по форме и обратно, имеет большое значение. Подобная автоматизация значительно ускоряет процесс разработки форм и структуры данных и позволяет избежать многих ошибок, которые может допустить пользователь при раздельном создании схемы данных и форм.

Автоматизацию создания форм или данных можно разделить на 4 уровня, каждый из которых включает в себя предыдущие:

0. Отсутствует;
1. По элементу данного можно создать элемент формы или по элементу формы создать элемент данного;
2. По схеме данных (или ее части) можно создать форму или имея форму можно создать схему данных;
3. Уровень 2 + автоматическая поддержка связей данных и элементов формы при редактировании + возможность сливания схемы данных и форм после редактирования схемы или формы.

Уровень автоматизации создания форм или схемы данных в рассматриваемых системах показан в табл. 2.

Гибкость настройки и наполнения словарей

Для ввода данных в форму очень часто пользователю предоставляется возможность ввода данных из словаря — готового набора возможных значений поля формы. Словари могут различаться по динамичности их наполнения и по способу выбора значений.

Динамичность словарей определяется временем их наполнения. Словари могут наполняться значениями в процессе работы с системой или только при редактировании словарей администратором системы:

1. Статические независимые;
2. Статические зависимые — в зависимости от заполненных данных, к полю подключается тот или иной статический словарь;
3. Динамические независимые — словари наполняются значениями в процессе работы с системой;
4. Динамические зависимые — словари наполняются значениями в процессе работы с системой и состав словаря может изменяться в зависимости от других значений, введенных в форму.

По способу выбора значений или по структурированности словари можно разделить на:

1. Одноуровневые — простой список возможных значений;
2. Одноуровневые многозначные — простой список возможных значений, каждое из которых может включать несколько значений, используемых для различных целей, например, одно значение показывается, а другое реально устанавливается в поле или при выборе одного значения, несколько других подставляются в другие поля;
3. Иерархические — дерево классифицированных значений;
4. Иерархические многозначные.

Гибкость словарей в рассматриваемых системах показана в табл. 2. Гибкость оценивается по двум параметрам: динамичность (1– 4) и структурированность (1– 4).

Типизирование данных формы

Некоторые из рассматриваемых систем поддерживают проверку типов данных при вводе их в форму и хранят данные соответствующих типов, некоторые используют только текстовые значения и проверяют только соответствие формата вводимой строки в соответствии с каким-то типом данных, а некоторые позволяют вводить все данные в произвольном текстовом формате. Поэтому по типизированию данных системы можно разделить на следующие группы:

- Нетипизированные — используются только строковые данные произвольного формата;

- Слаботипизированные — используются только текстовые данные, но предусмотрена возможность контроля формата текстовых данных;
- Сильно типизированные — данные хранятся разных типов и при вводе их в форму производится проверка соответствия типа данных вводимым данным.

Для классификации систем по типизированности также полезно учесть количество типов и форматов для сильно типизированных, и количество форматов для слаботипизированных систем. Типизированность систем приведена в табл. 2.

Пользовательские свойства элементов форм

Некоторые системы позволяют создавать пользователем дополнительные атрибуты элементов форм, не ограничиваясь тем набором свойств, который разработчики системы посчитали достаточным для пользователя. Свойства, добавляемые пользователем, не могут быть использованы для изменения логики обработки данных системой, но могут быть использованы в сценарии работы формы, описанном на скриптовом языке или при обработке данных формы и самой формы, экспортированных в другую систему. Наличие возможности использования пользовательских атрибутов указано в табл. 2.

Выводы

В заключение можно отметить, что наиболее гибкой и автоматизированной системой создания и использования экранных форм является НИКА-технология. В плане реализации возможностей элементов управления экранных форм, на первом месте стоят Microsoft VBA и Web-формы, хотя в остальных аспектах они проигрывают некоторым рассмотренным системам.

Таким образом, если пользователям требуются формы наиболее красивые и наполненные самыми разнообразными элементами управления, то им можно порекомендовать систему, использующую Microsoft VBA, или Web-формы. Если же требуются формы легко настраиваемые или использующие сложные или наращиваемые структуры данных, то наиболее подходящей системой является НИКА-технология.

Таблица 2

Возможности электронных форм существующих систем, использующих формы для ввода и отображения данных

Характеристика / Система	Полнота реализации возможностей элементов управления (%)	Структурированность вводимых данных (1–4)	Полнота реализации формул для полей форм (набор функций; количество атрибутов с формулами)	Уровень автоматизации взаимодействия форм и данных при создании формы или схемы данных (0–3)	Гибкость настройки и наполнения словарей (динамичность, структурированность)	Типизирование данных форм	Наличие возможности использования пользовательских атрибутов
Microsoft Word	31,6	2	(неуд.; 0)	0	(1, 1)	Нетипизир.	–
Microsoft VBA и диалоговые формы	68,4	1	(неуд.; 0)	0	(1, 1)	Нетипизир.	–
Microsoft InfoPath	73,7	4	(удовл.; 2)	1	(1, 1)	Сильнотипизир., 8 типов	–
Web-формы	68,4	2	(неуд.; 0)	0	(1, 1)	Нетипизир.	–
Lotus Notes	57,9	2	(хорошо; 4)	1	(1, 1)	Сильнотипизир., 8 типов	–
Adobe Acrobat	36,8	2	(удовл.; 1)	0	(1, 2)	Слаботипизир., 7 расширяемых форматов	–
Oracle Forms	52,6	3	(неуд.; 0)	2	(3, 1)	Сильнотипизир., 6 типов	
НИКА-технология	73,7	4	(хорошо; 5)	3	(4, 4)	Сильнотипизир., 18 расширяемых типов, произвольный формат	+

Литература

1. Microsoft Office Online: InfoPath 2003 Home Page // http://office.microsoft.com/home/office.aspx?assetid=FX01085792&CTT=6&Origin=ES790020011033.
2. *Холмогоров В*. Microsoft Office 2003 — предварительный обзор // http://www.getinfo.ru/article414.html
3. *Дубинко М.* (*Micah Dubinko*). XForms и Microsoft InfoPath // http://www.iso.ru/journal/articles/314.html
4. Cascading Style Sheets home page (CSS) / W3C Org // http://www.w3.org/Style/CSS/
5. Руководство пользователя Microsoft Word 2000.
6. Руководство по разработке форм Lotus Notes. http://www.cprt.spb.ru/AAT/myjournal.nsf/f5b2cbf2a827c0198525624b00057d30/4beaad0fa513fd66c3256c8c00443fa5!OpenDocument
7. W3C HTML 4.01 Specification // http://www.w3.org/TR/html4/
8. Руководство пользователя Lotus Domino Designer, версия 5, глава 5.
9. http://doc.notes.net/uafiles.nsf/docs/designer50/$File/App_Dev.pdf.
10. Lotus Notes 6 — краткое описание продукта производителем // http://www.ibm.com/ru/software/lotus/client/notes6.html.
11. PDF Reference, Second Edition. Adobe Systems Incorporated, 2000.
12. *Митилино С.* Будь в форме: InfoPath 2003 // Компьютерное Обозрение. № 41, 21–27 октября 2003 // http://itc.ua/15205
13. Oracle Forms / Кафедра Вычислительной техники Санкт-Петербургского Института Точной Механики и Оптики (Технический Университет) // http://cs.ifmo.ru/education/documentation/forms2000/index.shtml
14. *Апанасенко Е. В.* Использование Internet/Intranet технологий для организации доступа к базам данных // http://www.csu.ac.ru/students/works/0001/index.html.
15. ГЛОССАРИЙ.ru // http://www.glossary.ru. http://www.glossary.ru/cgi-bin/gl_sch2.cgi?RRywzqyzw:!kgtt:)
16. Lotus Notes и Domino R5. Энциклопедия пользователя. ДиаСофт, 2000.
17. *Мюллер Р. Дж.* ORACLE Developer/2000 Настольная книга пользователя. Лори, 1999.
18. *Богданов А. С., Емельянов Н. Е., Ерохин В. И., Скорняков В. А., Романов Б. Л.* НИКА-технология построения информационных систем // Организационное управление и искусственный интеллект / Сборник трудов ИСА РАН. Под ред. член-корр. РАН В. Л. Арлазарова и д. т. н. проф. Н. Е. Емельянова. М.: УРСС, 2003. С. 52–67.