ЗАДАЧА ИДЕНТИФИКАЦИИ ФИЗИЧЕСКИХ И ЮРИДИЧЕСКИХ ЛИЦ В ХРАНИЛИЩАХ ДАННЫХ ПРИ ПОИСКЕ «ПОДОЗРИТЕЛЬНЫХ» ФИНАНСОВЫХ ОПЕРАЦИЙ

А.П. Игнатенко, Н.Ю. Мелин
ТОВ "Ер-Джі-Дейта Україна"
e-mail: ktulhu@rgdata.ukrtel.net

Постановка задачи. Системы, позволяющие осуществлять своевременное выявление и предупреждение «подозрительных» финансовых операций – важный элемент борьбы с терроризмом и отмыванием «грязных» денег. Эти системы являются частью технологии подготовки управленческих решений в государственной и финансовой сфере. При их построении необходимо учитывать следующие функции:

а) сбор, организация и подготовка специальных данных для анализа в виде постоянно наращиваемого хранилища данных [1, 2];

б) собственно анализ как основа принятия решений.

Создание системы сбора и хранения данных, которые регистрируют и характеризуют различные виды деятельности организации, приводит к наполнению хранилища разнообразной информацией. Такая информация, как правило, является разнородной и многоуровневой, поэтому требует применения специальных средств анализа. Важной особенностью хранилища является и то, что единожды занесенные в него данные затем многократно извлекаются из него и используются для анализа. Отсюда вытекает одно из основных преимуществ использования хранилищ в работе организации – контроль за критически важной информацией, полученной из различных источников.

Наиболее уязвимым местом использования хранилища, с точки зрения анализа и последующего принятия решения, является корректность загруженных в него данных. Данные перед загрузкой в хранилище должны быть либо "очищены от шума", либо обработаны методами нечеткой логики, допускающей наличие противоречивых фактов. Другим подходом является ведение *общего хранилища*, в которое грузится вся пришедшая информация и *эталонного хранилища*, информация которого используется для анализа. Существуют различные методы обработки и фильтрации «грязной» информации и наполнения эталонного хранилища. Одним из таких методов обработки является идентификация данных.

В данной работе рассматривается задача идентификации экземпляров сущностей хранилища данных, описывающих физических и юридических лиц, однако подобные методы применимы и к другим типам сущностей. Такого рода задачи часто возникают при создании и ведении корпоративных хранилищ данных. Причинами их возникновения являются следующие факторы:

- Независимость источников – у каждой информационной системы существуют *собственные схемы хранения и правила ввода* данных. Кроме того, список хранимых атрибутов и полнота описания может существенно отличаться. В результате чего одни и те же сущности описаны разным набором атрибутов с различным составом и полнотой.
- Отсутствие или недостаточная проработанность процедур контроля при вводе и перегрузке данных. В итоге в хранилище происходит *умножение экземпляров* одного физического или юридического лица.
- Ошибки ввода – как показывает практика, *всегда* сопутствуют вводу данных в информационную систему.
- Наличие в хранилище данных, представление которых *допустимо различным образом* (формат, язык и т.п.). К примеру, фамилия физического лица может быть введена на русском, украинском или английском.

Решение этой задачи является достаточно нетривиальным и включает в себя как математические алгоритмы нечеткого анализа и кластеризации, так и создание вспомогательных структур хранилища, поддерживающих ведение эталонов и версий. Предлагается решать задачу в рамках создания системы сопоставления записей, которая должна обеспечивать загрузку новых данных, их анализ, идентификацию и запись в хранилище. Различные аспекты создания такой системы описаны в работе [3]. Ее ядром является подсистема идентификации лиц, реализующая математические алгоритмы [4] сравнения записей рабочего набора – множества не идентифицированных записей с эталонным набором. Разобьем процесс идентификации записей на несколько этапов:

1. Точное сравнение – первый этап. На этом этапе происходит точное сравнение значимых полей (т.е. одинаковых для разных реализаций, например дата рождения, идентификационный код, ФИО), описывающих элементы рабочего набора с элементами эталонного набора. В случае полного совпадения всех полей происходит автоматическое связывание двух записей и дальнейший анализ не проводится. Исключительные ситуации, такие как смена фамилии, происходят довольно редко и обрабатываются отдельно.
2. Нечеткое сравнение – второй этап. Здесь данные сравниваются с применением методов нечеткой логики. Эффективность данной стадии зависит от грамотно подобранных критериев идентичности, согласно которым и делается вывод об удачном сопоставлении. Разработка и уточнение этих критериев является одной из *самых сложных* задач при проектировании и эксплуатации подсистемы.
3. Ручное связывание – это последний этап. На этом этапе требуется участие человека. Собственно, на входе имеется список похожих пар "эталон-рабочий образец", а окончательное решение об их идентичности принимает оператор. Для взаимодействия оператора с системой необходимо разработать GUI программу, которая должна позволять сформировать набор связанных записей оператору, не обладающему навыками программирования.

Методы нечеткого сравнения. Исходя из описанного принципа работы подсистемы, легко заметить, что основной вклад в эффективность ее работы вносит именно стадия нечеткого сравнения. Информация, описывающая физическое или юридическое лицо, как правило, содержит следующие поля (табл. 1.)

Таблица 1.

	Физические лица	**Юридические лица**
1.	Идентификационный код	Код ЕДРПОУ
2.	Фамилия	Полное название
3.	Имя	Короткое название
4.	Отчество	-
5.	Дата рождения	Дата регистрации
6.	Паспортные данные	Регистрационные данные

Для оценки степени идентичности двух записей используются деревья решений [4]. Дерево решений – это структурированное множество правил, которое позволяет классифицировать данные по определенным критериям. Дерево решений является плоским графом (рис. 1.) . Каждый узел дерева соответствует некоторому варианту сравнения пары полей – совпадает, не совпадает, отсутствие данных. Прохождение по дереву тем или иным путем соответствует степени отдаленности рабочей записи от эталонной. При этом для полей 1,5,6 используется точное сравнение, а для полей 2 – 4 метод Q-грамм.

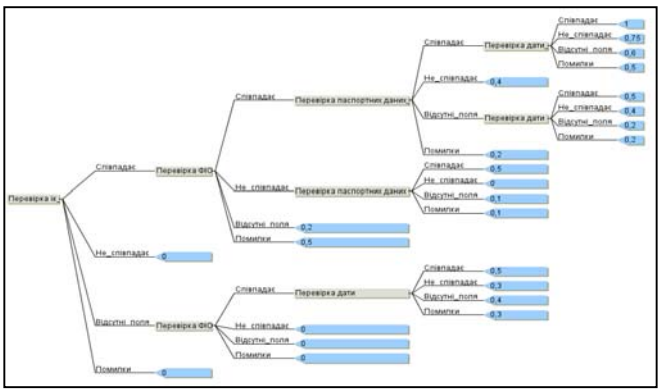

Рис. 1. Пример дерева решения

Метод Q-грам. Это один из методов сравнения текстовых строк [5], который показывает хорошие результаты для таких видов данных, как фамилии или названия фирм. Суть метода заключается в том, что сравниваемые строки режутся на подстроки длины Q (Q-грамы), далее осуществляется сравнение наборов подстрок и, исходя из количества совпавших элементов, делается вывод об их степени схожести.
Критериями, в данном случае, могут служить следующие величины:

- Количество совпавших подстрок с учетом длин сравниваемых строк;
- Количество совпавших подстрок с учетом их позиций в сравниваемых строках;
- Количество совпавших подстрок с учетом их позиций в сравниваемых строках и длин сравниваемых строк;
- Количество совпавших подстрок с учетом допустимого интервала отклонения их позиций в сравниваемых строках и допустимого отклонения длин сравниваемых строк.

Реализация. В качестве базовой технологии для разработки приложения выбрана технология Java – многоплатформенная переносимая объектно-ориентированная среда программирования. Система сопоставления записей была реализована на базе платформы Blaze Advisor 5.5 компании Fair Isaac (http://www.fairisaac.com). Архитектура приложения изображена на рисунке 2.

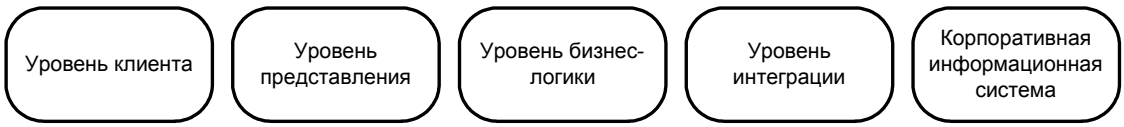

Рис. 2. Общая схема многоуровневого приложения

Многоуровневые приложения, как правило, одновременно являются многопользовательскими, поскольку предназначены для коллективного использования. Основным недостатком архитектуры клиент-сервер является то, что любые изменения в бизнес логике или функциях доступа к данным, вызывающие изменения в клиентской части приложения (в зависимости от «толщины» клиента), требуют переустановки новой клиентской части на всех рабочих местах. Поэтому клиентский уровень в нашем случае представлен тонким клиентом, за счёт чего достигается требуемая гибкость предоставления сервисов удалённым терминалам. Таким образом, клиентская часть отделена от большинства изменений в остальных частях приложения.

Уровень представления в нашем случае состоит из следующих программных компонентов:

- Java Server Pages;
- HTML pages;
- JavaScripts;

- Прочие файлы

Уровень бизнес-логики функционирует в адресном пространстве веб-сервера и содержит компоненты JavaBeans.

Уровень интеграции приложения с другими информационными системами осуществляется через соответствующие адаптеры ресурсов. На этом уровне реализуются соединения с хранилищами данных.

Blaze Advisor предоставляет широкие возможности по разработке бизнес уровня приложения. Средствами генерации компонент бизнес-логики можно получать полностью работоспособное приложение, готовое к использованию.

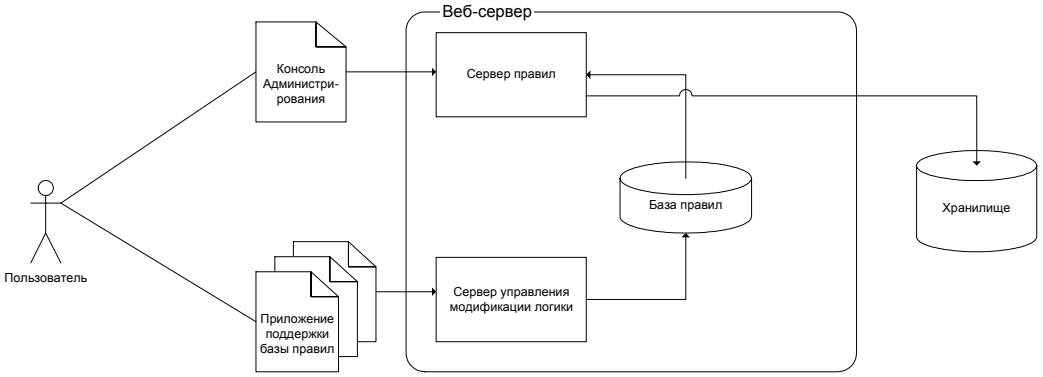

Рис. 3 Архитектура приложения

В результате было создано приложение, схема функционирования которого представлена на рисунке 3.

Пользователь работает с программным комплексом с помощью интерфейсов «консоль администрирования» и «приложение поддержки базы правил». Эти интерфейсы обращаются к веб-серверу, который организовывает работу приложения в соответствии с описанными правилами анализа. Основными элементами веб-сервера являются сервер управления модификации логики, база правил и сервер правил.

База правил хранит логику анализа информации хранилища. Сервер управления модификации логики позволяет изменять базу правил без перепрограммирования системы. Сервер правил обеспечивает запуск приложения, связь с хранилищем и анализ данных на основе базы правил.

Литература.

1. Э. Спирли, Корпоративные хранилища данных. Планирование, разработка и реализация. т. 1: Пер. с англ. – М.: Издательский дом «Вильямс», 2001. – 400 с.
2. И.Т.Кадощук, Е.А.Липчинский, Обзор технологии хранилищ данных. http://www.olap.ru/basic/genstore.asp
3. Д. Орлов, Подсистема сопоставления записей в хранилище данных, http://www.olap.ru/basic/CompareLog_dw.asp
4. Mueller J.-A., Lemke F. Self-Organizing Data Mining. Berlin, Dresden 1999.
5. Graham A. Stephen, String Search, Technical Report TR-92-gas-01 School of Electronic Engineering Science University College of North Wales Dean Street, Bangor, Gwynedd, UK LL57 1UT, October 1992, русский перевод: http://www.3ka.mipt.ru/vlib/books/Programming/ComputerScience/StryngAnalysis/index.html