**УДК 681.5**

# НОВАЯ ТЕХНОЛОГИЯ АВТОМАТИЗАЦИИ ИНТЕРНЕТ-ОПЕРАЦИЙ

***канд. техн. наук Д.О. ГЛУХОВ, Т.М. ГЛУХОВА***

*Представлена новая технология сбора и манипулирования данными, основанная на применении абстрактного робота, конкретизируемого в каждом отдельном случае соответствующей программой на разработанном нами формальном языке манипулирования информационными структурами Интернет. Язык основан на разрабатываемой нами алгебре, замкнутой относительно коллекций деревьев.*

*Приводится пример робота, коллекционирующего в базе данных ссылки, найденные по некоторому запросу поисковой системой Yandex. Пример иллюстрирует возможности робота по управлению поисковыми системами.*

## 1. Формализация

### 1.1. Основные определения

Интернет - это распределенный текст. Устаревший и вместе с тем наиболее точный с прагматической точки зрения тезис. Этот текст структурирован языком HTML, однако представляет собой текст на естественном языке. Структура текста взаимосвязана с текстом и служит его оформлением. Современный взгляд на вопросы структурирования связан с все большей формализацией гипертекста и разделением функций.

Так, на смену HTML пришел XHTML, который все чаще заменяется комбинацией XML-XSLT.

Исторически сложилось, что текст в Интернет разбит на такого рода документы. Каждый документ «читается» как DOM (Document Object Model) структура. Эта структура в терминологии DOM модели, разработанной W3C консорциумом и принятой как рекомендация, представлена в объекте класса HTMLDocument [1].

Формально HTMLDocument представляет собой лес. Древовидные структуры достаточно хорошо формализованы и изучены. Свойства таких систем рассматриваются в большом количестве публикаций. Интерес к ним был особенно острым на заре становления теории баз данных, при разработке иерархических СУБД (систем управления базами данных). На современном этапе развития информационных технологий к таким структурам опять возвращаются, говоря о таких технологиях, как XML, ОРСУБД (объектно-реляционные системы управления базами данных), ООСУБД (объектно-ориентированные) [3 - 6].

Касательно Интернет-структур, документ есть множество деревьев:

$$HTMLDocument = E = \{ X, R, R_{seq} \}, \tag{1}$$

где каждый элемент, согласно стандарту, имеет следующую структуру:

$$X = \{ Element_i \tag{2}$$

$$Element = < A, V, R_{av} >,$$

где $A$ – множество атрибутов; $V$ – множество значений атрибутов; $R_{av}$ – отношение множества атрибутов ко множеству значений атрибутов, такое, что $\forall a \in A \; ! \exists v \in V | \; <a,v> \; \in R_{av}$.

Для удобства записи введем оператор взятия значения:

$$x.a : \; v \mid x \in X, \; a \in x.A, \; v \in V \wedge <a,v> \in R_{av}$$

Узел характеризуется парами объектов – означенными атрибутами. Атрибут не может иметь два значения одновременно.

Отношение $R$:

$$R = X \times X \tag{3}$$

является бинарным отношением вложенности, транзитивным, асимметричным, антирефлексивным отношением на множестве $X$.

А отношение $R_{seq}$:

$$R_{seq} = X \times X \tag{4}$$

является отношением следования одноуровненных элементов, имеющих общий родительский узел. Данное отношение также является транзитивным, асимметричным, антирефлексивным отношением на множестве $X$ [5, 6].

Стартовой вершиной последовательности является вершина:

$$start(R_{seq}) = X \setminus \bigcup_{x \in X} R_{seq} / \{x\}. \tag{5}$$

### 1.2. Теория множеств и теория графов

При формализации выбранной предметной области нами эксплуатируется теория множеств и теория графов. Используемые в данной статье операции над множествами:

$$A \cup B := \{x \mid x \in A \vee x \in B\},$$
$$A \cap B := \{x \mid x \in A \wedge x \in B\},$$
$$A \setminus B := \{x \mid x \in A \wedge x \notin B\},$$
$$A / B := \{x \mid \forall b \in B \; \exists <x, b> \in A\}$$

заимствованы из реляционной алгебры [2]

$$A \times B := \{<a, b> \mid a \in A \wedge b \in B\}.$$

*Дополнительные обозначения*

Множество $E$ будем рассматривать как несвязанный ациклический орграф без петель.

Поскольку $E$ является орграфом, то транзитивным замыканием $R^{+}$ на $X$ будет отношение достижимости на орграфе $E(X, R)$ [3].

Инволюцию отношения $R$ будем обозначать $(R)^{-}$

Тогда поддеревом леса $E$, полученным одной из компонент связности исходного леса по заданному корневому узлу $a$, будет называться дерево:

$$E_a(X', R', R'_{seq}) \quad \left| \begin{array}{l} X' = ((R^{+})^{-} / \{a\}) \cup \{a\} \\ R' = R \setminus [R \cap ((X \setminus X') \times X)] \\ R'_{seq} = R_{seq} \setminus [R_{seq} \cap ((X \setminus X') \times X)]. \end{array} \right. \tag{6}$$

Определим оператор вычленения поддерева:

$$subtree(E, a) = E_a. \tag{7}$$

Определим оператор выбора дочерних узлов, непосредственно вложенных в узел $a$:

$$childrenOf(E, a) = \{x \mid x \in ®^{-} / \{a\}\}$$

и всех связанных с $a$ узлов:

$$allChildrenOf(E, a) = \{x \mid x \in (R^{+})^{-} / \{a\}\}.$$

### 1.3. Отношение эквивалентности узлов по атрибутам

Определим бинарное отношение эквивалентности по атрибутам на множестве узлов. Отношение эквивалентности обладает свойствами транзитивности, рефлективности, симметричности:

$$x_1 = x_2 := \{<x_1, x_2> \mid \forall a \in x_1.A \cup x_2.A \; <a, v> \in x_1.R_{av} \wedge$$
$$<a, v'> \in x_2.R_{av} \Rightarrow v = v'\}. \tag{8}$$

В одном дереве может быть множество эквивалентных узлов и соответственно классов эквивалентности узлов.

Сформулированное отношение требует, чтобы узлы имели в точности совпадающие множества атрибутов, означенных одинаково. Если предположить, что узлы имеют различные множества атрибутов, то появляется необходимость рассмотреть различные способы сопоставления таких узлов.

### 1.4. Отношение нестрогого порядка на множестве узлов по атрибутам

Определим бинарное отношение порядка по атрибутам на множестве узлов. Отношение порядка обладает свойствами рефлективности, антисимметричности и транзитивности. Очевидно, что мы можем предложить множество отношений, обладающих такими свойствами. Но мы построим отношение нестрогого порядка, имеющее прагматичную интерпретацию. Вводимое отношение должно показывать совпадение узлов с заданным шаблоном:

$$x_1 < x_2 := \left\{< x_1, x_2 > \middle| x_1.R_{av} \subseteq x_2.R_{av}\right\}.$$

Данное отношение устанавливает подобие узла $x_2$ узлу $x_1$. Устанавливает, что узел $x_2$ включает узел $x_1$, но имеет возможность быть шире.

### 1.5. Отношение эквивалентности узлов по ссылкам

Определим бинарное отношение эквивалентности по ссылкам на множестве узлов:

$$x_1 \rho x_2 := \{ < x_1, x_2 > \mid x_1 \equiv x_2 \}. \tag{9}$$

Узел одного дерева эквивалентен по ссылке узлу другого дерева, если это один и тот же узел. Это означает, что узел, может входить одновременно в различные деревья. Такое отношение эквивалентности более ограничительно отношения эквивалентности по атрибутам. Любые два узла, эквивалентные по ссылке эквивалентны по атрибутам, но не наоборот.

### 1.6. Отношение нестрогого порядка на множестве поддеревьев

Вводя отношения порядка на множестве деревьев, мы преследуем цель получения эффективного механизма распознавания поддеревьев подпадающих под заданную структуру. Пусть задан лес $E(X, R, R_{seq})$. Поддерево имеет две важные особенности:

- во-первых, оно имеет корень – узел $a$, и определяется как

$$subtree(E, a);$$

- во-вторых, количество компонент связности поддерева

$$k(subtree(E, a)) = 1,$$

т.е. оно представляет собой связанный орграф.

Формулируя интерпретацию конструируемого отношения, можно выделить два варианта распознавания деревьев:

- без учета порядка следования подузлов в пределах одного уровня вложенности;
- с учетом порядка следования дочерних узлов одного уровня вложенности.

Каждый из этих вариантов может иметь возможность распознавания структуры с непосредственной вложенностью узлов или необязательно непосредственной, а некоторой возможной опосредованной вложенностью.

### 1.7. Отношение нестрогого порядка без учета последовательностей на множестве поддеревьев

$$subtree(E, a_1) < subtree(E, a_2) := \left\{ < subtree(E, a_1), subtree(E, a_2) > \;\middle|\; \begin{array}{l} a_1 < a_2 \\ \forall x \in childrenOf(E, a_1) \\ \exists y \in childrenOf(E, a_2) \\ такой, \quad что \\ subtree(E, x) < subtree(E, y) \end{array} \right\} \tag{10}$$

Расширим определение узла, введя дополнительный атрибут шаблона:

$$X = \{ Element_i \}, \tag{11}$$

$$Element = < A, V, R_{av}, isParent >.$$

Атрибут *isParent* показывает, что определенные дочерние узлы принадлежат данному родительскому узлу непосредственно, а не просто присутствуют во вложенности. Если он *false*, то, как раз наоборот, дочерние узлы присутствуют во вложенности, а не непосредственно связаны с ним. Тогда можно расширить и определение отношения порядка с учетом снятия ограничения по непосредственной вложенности:

$$subtree(E, a_1) < subtree(E, a_2) := \left\{ < subtree(E, a_1), subtree(E, a_2) > \;\middle|\; \begin{array}{l} a_1 \prec a_2 \\ \forall x \in childrenOf(E, a_1) \\ \exists y \in childrenOf(E, a_2) \mid a_1.isParent \\ \exists y \in allChildrenOf(E, a_2) \mid otherwise \\ такой, \quad что \\ subtree(E, x) < subtree(E, y) \end{array} \right\} \tag{12}$$

Таким образом, мы получаем возможность анализа совпадения с шаблоном как фрагмента дерева, так и фрагмента «разорванного» по вложенности.

### 1.8. Отношение нестрогого порядка с учетом последовательностей на множестве поддеревьев

Введем отношение включенности последовательностей:

**Утверждение 1.** Последовательность, заданная отношением следования одноуровненных элементов $R_1$, сохраняется в последовательности $R_2$, если

$$\forall < x_1, x_1' > \in R_1 \; \exists < x_2, x_2' > \in R_2^+ \;\; \Rightarrow \;\; x_1 = x_2 \wedge x_1' = x_2' . \tag{13}$$

***Доказательство:***

1. *Необходимость. От противного.*

Пусть даны отношения $R_1 = X_1 \times X_1$ и $R_2 = X_2 \times X_2$, и $X_1 \subset X_2$.

Предположим, что последовательность не сохраняется и существует пара $< x_1, x_2 > \in R_1$ такая, что в транзитивном замыкании отношения $R_2$ присутствует пара $< x_2, x_1 >$.

Из антирефлексивности отношения $R_2$ следует, что его транзитивное замыкание, обладающее свойством минимальности, также антирефлексивно:

$$\exists < x_2, x_1 > \in R_2^+ \;\Rightarrow \exists < x_2, x_1 > \in R_2^+ \;\Rightarrow \neg \exists < x_1, x_2 > \in R_2^+ .$$

2. *Достаточность.* По индукции, поскольку более сложные нарушения последовательности можно представить как результат простых перестановок. Все возможные комбинации последовательностей ограничиваются числом $(|R_1|+1)!$.

Утверждение 1 сформулировано для заданного на множестве узлов отношения эквивалентности $a = b$. Обобщим утверждение 1 на произвольное отношение порядка на $X$. И будем говорить, что последовательность, заданная отношением следования одноуровненных элементов $R_1$ на множестве элементов $X_1 \subset X$, при заданном на $X$ отношении порядка узлов сохраняется относительно данного свойства, если

$$\forall < x_1, x_1' > \in R_1 \; \exists < x_2, x_2' > \in R_2^+ \;\; \Rightarrow \;\; x_1 < x_2 \;\; \wedge x_1' < x_2' . \tag{14}$$

Для учета сохранения порядка следования узлов по отношению порядка узлов на поддеревьях необходимо утверждение 1 по индукции расширить на последовательность поддеревьев. Определение отношения порядка на поддеревьях при этом примет рекурсивный вид:

$$subtree(E, a_1) < subtree(E, a_2) := \left\{ < subtree(E, a_1), subtree(E, a_2) > \;\middle|\; \begin{array}{l} \forall < x_1, x_1' > \in R_{seq_1} \; \exists < x_2, x_2' > \in R_{seq_2}^+ \\ subtree(E, x_1) < subtree(E, x_2) \\ \wedge subtree(E, x_1') < subtree(E, x_2') \\ \wedge x_1 < x_2 \\ \wedge x_1' \prec x_2' \end{array} \right\} \tag{15}$$

Рассматривая введенное ранее расширение узла (11), определим условие совпадения следующим образом:

$$subtree(E, a_1) < subtree(E, a_2) := \left\{ < subtree(E, a_1), subtree(E, a_2) > \;\middle|\; \begin{array}{l} \forall < x_1, x_1' > \in R_{seq_1} \; \exists < x_2, x_2' > \in R_{seq_2}^+ \\ \wedge \forall < x_1, x_1' > \in R_{seq_1} \; \exists < x_2, x_2' > \subset \bigcup\limits_{allChildrenOf(a_2)} R_{seq}^+ \;\middle|\; \neg a_1.isParent \\ subtree(E, x_1) < subtree(E, x_2) \\ \wedge subtree(E, x_1') < subtree(E, x_2') \\ \wedge x_1 < x_2 \\ \wedge x_1' < x_2' \end{array} \right\} . \tag{16}$$

Последний вариант наилучшим образом подходит для шаблонного поиска в Интернет структурах, поскольку учитывает как последовательность, обычно априори известную и неизменяемую, так и возможность «разорванности» по вложенности шаблона в структуре документа.

### 1.9. Ссылочный принцип

Отношение порядка на множестве поддеревьев устанавливает принцип сравнения поддеревьев, на предмет их неатрибутной идентичности. Однако все операции над множествами поддеревьев определены посредством отношения эквивалентности, которое определяет ссылочную эквивалентность поддеревьев. Иными словами, два поддерева будут идентичны тогда и только тогда, когда они есть суть одно и то же дерево, входящее в состав разных деревьев как поддерево.

Этот принцип реализован в языке *Java* и *C#* при сравнении на равенство объектов. В этих языках рассматривается и атрибутивное отношение эквивалентности по методу *Equals()*. Реализация таких теоретико-множественных операций запланирована в следующих версиях робота.

## 2. IDAMAR- новая технология автоматизации ИНТЕРНЕТ-операций

IDAMAR расшифровывается как (Internet Data Acquisition and Manipulation Abstract Robot) абстрактный робот сбора и манипулирования данными Интернет.

### 2.1. Особенности

Особенности IDAMAR:

- рассматривает наполнение сети как единую базу данных, в которой хранятся деревья с информационным наполнением;
- использует шаблонный поиск деревьев;
- манипулирует найденными древовидными структурами;
- интегрируется с реляционной базой данных для фиксации результатов обработки контента;
- помечает посредством шаблона узлы деревьев именами, для быстрого доступа к информации;
- позволяет перемещаться по сети известной структуры по заданному алгоритму;
- поддерживает циклические и условные переходы по сети;
- легко интегрируется с Java-приложением посредством нотификаций, позволяет создавать сложную обработку нотификаций в Java-программе, таких как перенос данных в реляционные базы данных после вторичной обработки;
- поддерживает Proxy авторизацию;
- позволяет формировать активный алгоритм, выполняющий помимо сканирования функции регистрации, заполнения форм и отправки данных форм методом POST;
- список поддерживаемых кодировок текста определяется стандартным модулем Java il8n.jar.

### 2.2. Структура IDAMAR

Конструктор класса RobotParser получает при создании текст программы, компилирует ее в объект интерпретатор и, в дальнейшем, инкапсулирует в себе объект интерпретатор. Java программа, использующая данного робота, может запустить его на интерпретацию программы командой *interpretQ*. Программа Java может также установить блоки кода, который будет вызван роботом в тех или иных случаях, как показано на рис. 1. С целью эффективной реализации системы применен архитектурный шаблон Адаптер [7, 8].

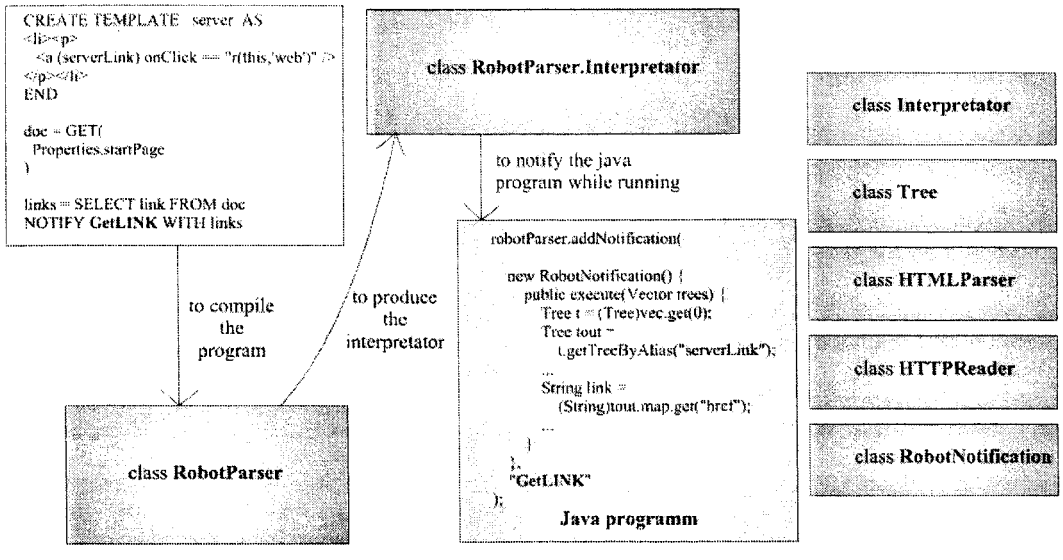


Рис. 1. Структура робота, формируемого по технологии IDAMAR

Представление содержимого сети возложено на класс Tree. Данный класс описывает дерево и допустимые над ним операции. Если быть более точным, то речь идет не об отдельном дереве, а о множествах деревьев. Все операции, определенные в классе Tree, параметрами и результатом рассматривают множества деревьев.

Пример интегрированного Java-проекта иллюстрируется на рис. 2.

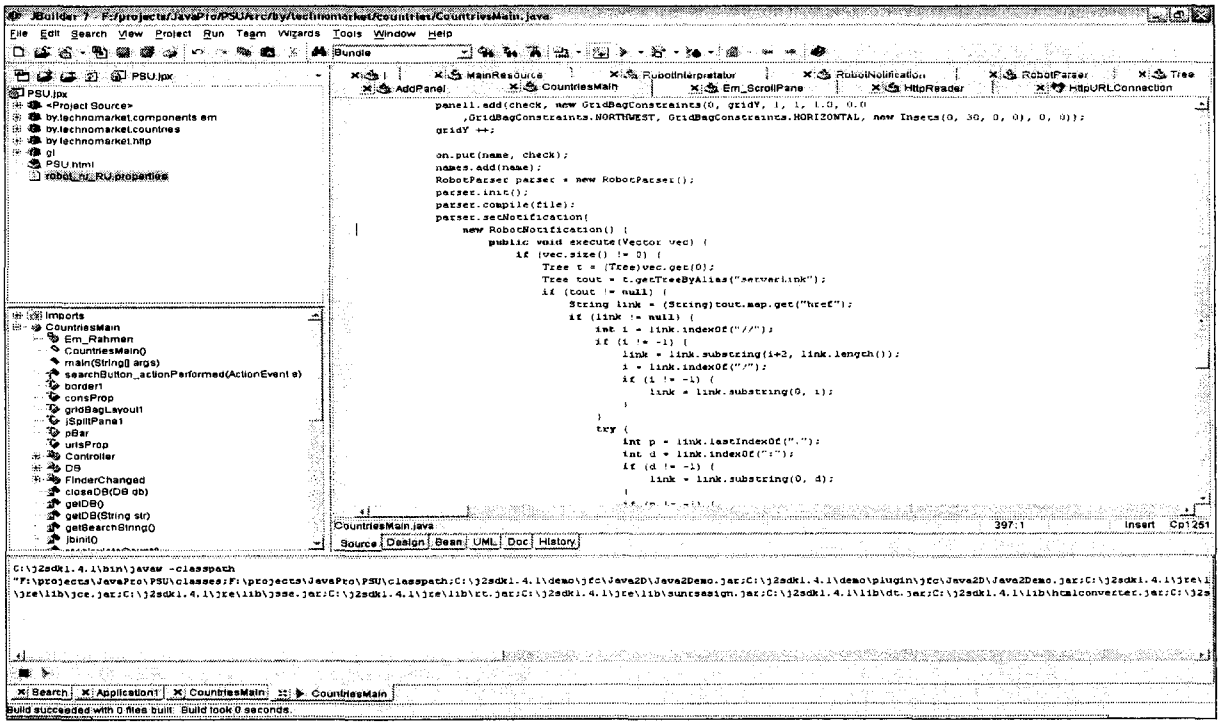

Рис. 2. Интеграция робота с Java-приложением

## 3. Реализация языка манипулирования информационными структурами ИНТЕРНЕТ

Определим класс деревьев и функции шаблонного поиска и выборки по шаблону поддеревьев по отношению вхождения, определенному в (15).

### 3.1. Множество поддеревьев

Очевидно, что вершиной $a$ возможного поддерева дерева $E(X, R)$ может являться только такой узел, для которого выполняется условие

$$(\{a\} \times X) \cap R \neq \varnothing, \text{ то есть, } \exists <a,b> \in R.$$

Тогда можно построить множество узлов $ROOTS$ допустимых вершин поддеревьев дерева $E$:

$$ROOTS = \bigcup_{x \in X} R / \{x\}. \tag{17}$$

Тогда множество всех поддеревьев $SUBTREES$ определяется как

$$SUBTREES = \{ subtree(E, a) \mid a \in ROOTS \}. \tag{18}$$

### 3.2. Язык манипулирования множествами деревьев

Для задания языка программирования роботов использована БНФ (Бекуса-Наура форма) нотация описания формальной грамматики языка [12].

Нами построена КС (контекстно-свободная) грамматика и выбран метод синтаксического анализа LR (1) [13].

Восходящий LR (1) синтаксический анализатор языка, заданного контекстно-свободной грамматикой, транслирует текст программы в программный объект интерпретатор. Порождаемый объект может быть запущен на интерпретацию многократно при необходимости, а также может быть сериализован на диск с целью хранения в бинарном варианте для его дальнейшего использования.

Рассмотрим грамматику более подробно.
Программа для робота представляет собой последовательность операторов:

```
Programm ::=
        | Operator
        |  Programm Operator
```

Допустимы операторы присваивания, системные, определения шаблонов, навигационные:

```
Operator ::=
 NAME "=" Set
        | "SET PROXY AUTHENTIFICATION" "("
"STRING" ","
"STRING" ","
"STRING" ","
"STRING"
 ")"
        | Template
        | Navigation
  | Set "." NAME "=" Value
  | Sql
```

Интеграция с базой данных позволяет конструировать любые запросы в базу данных на основе результатов работы с коллекциями деревьев. В настоящий момент не поддерживается управление транзакциями.

```
Sql ::= "SQL" SQLSEQUENCE

SQLSEQUENCE ::=        STRING
                     | Value
                     | SQLSEQUENCE "+" STRING
                     | SQLSEQUENCE "+" Value
```

Язык манипулирования поддеревьями замкнут относительно множества деревьев:

```
Set ::=      NAME
        | Set "UNION" Set
        | Set "INTERSECT" Set
        | Set "SUBTRACT" Set
| "GET" "(" Url ")"
        | "GET" "(" Url "," Encoding ")"
| "POST" "(" Url "," Set ")"
| "SELECT" Set "FROM" Set
        | NAME "(" NAME ") "
        | Sql
Value ::= STRING
      | Set "." NAME
```

Область описания шаблона служит для формулировки дерева, с которым отношением нестрогого порядка (отношением совпадения с шаблоном) будут сравниваться деревья при выполнении шаблонного поиска.

```
Template ::= "CREATE TEMPLATE" NAME "AS" TAG

TAG ::=            OPENTAG TAGS CLOSETAG
                 | OPENTAG TEXTPATTERN CLOSETAG
| "<" NAME ATTRSET "/>"
| "<" NAME "(" NAME ")" ATTRSET "/>"
TEXTPATTERN ::= STRING
TAGS ::=   TAG
      | TAGS TAG
OPENTAG ::=   "<" NAME ATTRSET ">"
                 | "<" NAME "(" NAME ")" ATTRSET ">"
           | "<...>"
ATTRSET ::=   ATTRIBUTE
        | ATTRSET ATTRIBUTE
```

На атрибуты можно накладывать разнообразные условия вида:

```
ATTRIBUTE ::=   NAME "==" STRING
              | NAME "!=" STRING
              | NAME ">" STRING
              | NAME "<" STRING
              | NAME ">=" STRING
              | NAME "<=" STRING
              | NAME "LIKE" STRING
```

Причем операторы сравнения == и != имеют ссылочную интерпретацию отношения эквивалентности, операторы >, <, >=, <= рассматривают значения как числа типа Integer языка java и оператор LIKE задает регулярное выражение, которому должна соответствовать строка – значение атрибута:

```
CLOSETAG ::=    "</" NAME ">"
              | "</...>"
Navigation ::= WHILE "(" Set "IS NOT EMPTY" ")"
"{"
Programm
"}"
              | IF "(" Set "IS NOT EMPTY" ")"
"{"
Programm
"}"
              | FOREACH NAME IN Set
"{"
Programm
"}"
```

На рис. 3 представлено главное окно программы, которая по некоторому поисковому запросу накапливает статистику результатов поиска 10-ю разными поисковыми серверами, сортируют ссылки по 254-м различным странам. Тем самым показывая, в какой стране данной тематике уделяется большее внимание.

Пример программы для робота, перебирающей и коллекционирующей все страницы поиска поискового сервера yandex.ru на запрос «robot programming example», приведен далее.

В программе автоматизировано управление 10-ю поисковыми серверами, а именно: Yandex, Google, Yaho, Altavista, Aol, Lycos, Msn, Search, Alltheweb, Webcrawler.

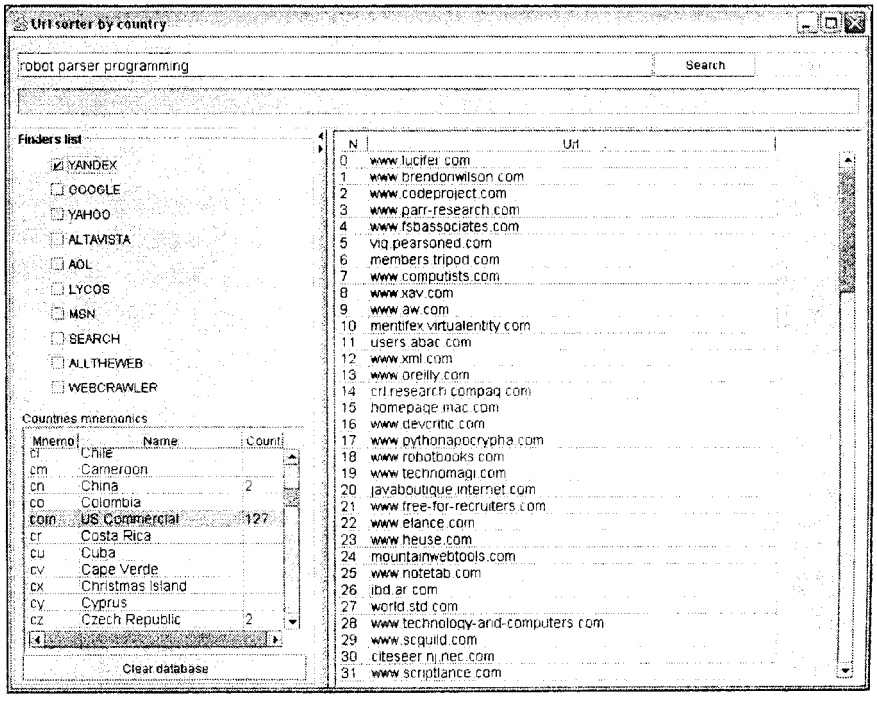


Рис. 3. Главное окно программы коллекционирования ссылок, найденных 10-ю поисковыми серверами, с сортировкой по странам размещения

### 3.3. Пример программы управления поисковым сервером Yandex

```
/**
 *          Dzmitry Hlukhau examples.
 *          Internet Robot Language (IRL)
 *
 *          Copyright (c) Polotsk State University 2005
 *
 *          Управление поисковым сервером Яндекс (www.yandex.ru)
 *
 */

//------------------------------ Прокси авторизация
SET PROXY AUTHENTIFICATION  (
“proxy.psu.by”,
“8080”,
“d.gluhov”,
“password”
)

  //------------------------------ Шаблоны фрагментов
CREATE TEMPLATE   next  AS
          <a (Next)  target LIKE “_self’>
                    “.*Дальше.* ”
          </a>
END

CREATE TEMPLATE   server  AS
          <li>
                    <p>
                              <a (serverLink) onClick == “r(this,'web') ” />
                    </p>
          </li>
END

CREATE TEMPLATE   count  AS
          <p>
                    <b (Total)>
                              “ [0-9]+ ”
                    </b>
          </p>
END

  //------------------------ Первичный запрос на поиск
doc =  GET(“http://yandex.ru/yandsearch?text=” +
Properties.searchString ,
“Cp1251”)

  //------------------- Коллекционирование найденных страниц
RefTag = doc

NOTIFY TOTAL WITH SELECT count FROM doc

WHILE ( RefTag   IS NOT EMPTY ) {
          RefTag = SELECT next FROM doc
          links = SELECT server FROM doc
          FOREACH link IN links { NOTIFY REFERENCE WITH link }
          doc =  GET( RefTag( Next ).href, “Cp1251”)
}
```

### 3.4. Перспективы дальнейшего совершенствования технологии

В проекте запланированы следующие задачи:

1. Создание среды программирования роботов с поддержкой режимов отладки, контекстных подсказок, подсветки кода (Robots Framework).

2. Разработка эффективной системы управления обработкой ошибок как на стадии трансляции, так и времени выполнения (Exception Control).

3. Реализация управления транзакциями и совершенствование интеграции с реляционными базами данных (Databases Support).

4. Развитие функциональных возможностей робота, разработка и оптимизация по скорости алгоритмов манипулирования древовидными структурами. Реализация высокоскоростных теоретико-множественных операций на базе атрибутивного отношения эквивалентности (Background Research).

5. Разработка мультизадачного сервера автоматизации, принципов коммуникации роботов при коллективном решении задач автоматизации (Internet Automation Server).

Предполагается, что робот, выступая в качестве агента сети, может применяться для решения задач, требующих коммуникаций между роботами. Разработка механизмов управления коллективами роботов является интересной и актуальной задачей [12, 13, 14].

## ЛИТЕРАТУРА

1. Document Object Model (DOM) Level 1 Specification / W3C Recommendation. - REC-DOM-Level-1-19981001. - Version 1.0: http://www.w3.org/TR/REC-DOM-Level-l, 1 October, 1998.
2. Ульман Д. Основы систем баз данных. - М.: Финансы и статистика. - 1983. - 335 с.
3. Новиков Ф.А. Дискретная математика для программистов. - СПб.: Питер, 2001. - 304 с.
4. Харари Ф. Теория графов. - М.: Мир, 1973.
5. Донской В.И. Дискретная математика. - Симферополь: СОНАТ, 2000. - 360 с.
6. Математические основы цифровой техники / В.М. Муттер, В.В. Трофимов, И.В. Иванова, М.Ю. Калинушкина. - СПб.: Литера плюс, 1999.- 351 с.
7. Cummings B. Object Oriented Analysis and Design with Applications Grady Booch. - Yourdon Press, 1994.
8. Object Oriented Analysis / C. Coad, et. al. - Yourdon Press, 1990.
9. Durfee E.H., Lesser V.R., Corkill D.D. Cooperative Distributed Problem-solving // In The Handbook of Artificial Intelligence. - 1989. -V. IV. - P. 83 - 148.
10. Chapman. D. Planning for Conjunctive Goals // Artificial Intelligence Journal, 1987. - № 32. - P. 333 - 367.
11. Barraquand. J., Latombe, J.-C. Robot Motion Planning: A Distributed Representation Approach // Research Report. - STAN-CS-89-1257. - Stanford: Department of Computer Science. Stanford University, 1989.
12. Гордеев А.В., Молчанов А.Ю. Системное программное обеспечение. - СПб.: Питер, 2001.-736с.
13. Stephen C.J.Y. Yet Another Compiler-Compiler. - New Jersey: Bell Laboratories Murray Hill, 2001.
14. Серебряков В.А., Галочкин М.П. Основы конструирования компиляторов. - М.: Эдиториал УРСС, 2001.-222 с.