"Свободное программное обеспечение в высшей школе",
Переславль-Залесский, 2010 г.

Динамический Рефал - инструмент для обучения сентенциальному программированию.

Исламов Марат Шамилевич,
Удмуртский государственный университет.

Задача ВПО: выпуск специалистов, способных быстро адаптироваться под изменяющиеся условия ИТ-индустрии

Всё большее осознание того, что ООП – не панацея

Нужен базис из языков разных стилей (парадигм)

Курс «Языки программирования» УдГУ:
Lisp (common), **Refal** (5), Prolog (SWI-Prolog)

Рефал в обучении

Алгоритмы Маркова

Шаблоны и сопоставление

Декларативный подход

Практические задания

“Антиспам”,
Релевантность слов (”Google”),
Парсеры (Pascal, C++, basic),
Транслятор (подмножество Pascal) → C
и др.

Проблемы

IDE

Программирование RegExp

Читаемость сложного кода

Цель

- расширить возможности управления сопоставлением в Рефале с помощью декларативных конструкций
- предоставить пользователю механизм абстракции данных с помощью пользовательских типов переменных
- все расширения должны быть концептуально непротиворечивы Базисному Рефалу

Динамический Рефал

Наследник Базисного Рефала

Программа = множество функций
Функция = список предложений
Предложение =
    {образец, результатное выражение}

Понятия: рефал-машина, объектное выражение,
структурные и функциональные скобки,
конкретизация, сопоставление, подстановка

Пример программы на Базисном Рефале

```
/* Замена A на B*/

ZamenaAB {
   'A' e.X      =   'B'  <ZamenaAB  e.X>;
   s.Y e.X      =   s.Y  <ZamenaAB  e.X>;
   (e.Y) e.X    =   (<ZamenaAB e.Y>)
                     <ZamenaAB e.X>;
   /* empty */ = /* empty */;
}
```

Типы данных Динамического Рефала

- Текстовый символ '***a***'
- Текстовый терм (compound symbol) ''***word***''
- Целое число ***123***
- Вещественное число ***123.12***
- Байт ***\x0A***

\+ структурные скобки = объектное выражение

Переменные

- **s**, **t**, **e** — переменные (Базисный Рефал)
- **E**-переменная — жадная **e**-переменная
- integer, real, alpha, byte
- **@** - обозначение для закрытых переменных
- Безымянная переменная

E. *'REFAL-'* **s**.version **e**. = **@**.version ;

Спецификаторы

Отсутствие спецификаторов — один из главных недостатков языка

- Группы
- Альтернативы
- Повторители

Группа - описание переменной, тип которой определяется некоторым образцом

Группа ::= **{** образец **}**

e. **{** *‘<Tag>’* **e**. *‘</Tag>’* **}**.tag **e**. @.tag **E**.

Пример. Образец с группой

Альтернатива - группа, имеющая несколько определяющих образцов (вариантов)

Альтернатива ::=

{ образец **|** образец **|** ... **|** образец **}**

{ '/*' **e**. '*/' | '//' **e**. '\n' | '\n*' **e**. '\n' }.*comnt* **E**.*other*

Пример. Образец с альтернативой

Повторитель — квантификатор повторения типа

Переменная с повторителем ::=
ОписаниеТипа[целое**..**целое **]**.*имяпеременной*

Пример: {'X'|'Y'|'Z'}[2..4].*variant*

“Целое” - может быть закрытой integer-переменной

Сокращения:
ОписаниеТипа[целое **...**]
ОписаниеТипа[целое]

Рефал-условие

Рефал-условия впервые появились в Рефале-4 и доказали свою состоятельность во всех последующих диалектах.

Условие ::=
РезультатноеВыражение **:** *Образец*

$NOT – отрицание условия!

```
s.nodigit,
    $NOT  '1234567890' : e.  s.nodigit  e.  ;
```

Пользовательские типы данных (пользовательские шаблоны)

Шаблон ::=

Template *имя* **::=** *описание* **;**

описание ::= *образец* { *условие* }*

Template GoodBrackets **::=**

{ */*empty*/* **|** *'('* GoodBrackets. *')'* GoodBrackets. **}**

Пример. Шаблон для списка правильно расставленных скобок

Доступ к внутренним переменным пользовательского шаблона

закрытая_переменная :: *вложенная*

- тоже закрытая переменная! => снова можно применить операцию «::»

```
Template HtmlDoc ::=
    '<html>' HeadCode.head  BodyCode.body '</html>';

Template HeadCode ::=
    '<head>' TitleCode.title '</head>' ;

Template TitleCode ::=
    '<title>' e.text '</title>' ;

Go {
    /* empty */,
        <ReadAllFile 'index.html'> :  HtmlDoc.doc,
        HtmlDoc.doc::head::title::text : 'Главная страница'
                        =  <Prout HtmlDoc.doc::body>;

        E.else = <Prout 'нет'>;
}
```

Программа находит в тексте теги, содержащие вложенные теги

```
Go { = <Run <Card>>;}

Template Tags ::=
    '<' e.word '>' e.body '</' e.word '>' ;

Template OwnTags ::=
    Tags.this,
    Tags.this::body : e. Tags.inside e. ;

Run {
    e. OwnTags.tt E.otherpart =
                <Prout @.tt::this::word> <Run @.tt::this::body>
                <Run E.otherpart>;
    E.else = <Prout 'no found'>;
}
```

Реализация. Не сделано

- Расширение языка
- синтаксический анализатор и загрузчик (на D-Refal)
- спецификатор повторения: убывание [10..3]
- спецификатор повторения: использование закрытых integer-переменных [integer.from .. 100]
- вещественные числа
- поддержка юникода (ICU)

СПАСИБО ЗА ВНИМАНИЕ

Информация:
http://ulm.udsu.ru/~soft/wiki/doku.php?id=d-refal:index

svn:
http://refal51.googlecode.com