

Лекция 1. Обзор возможностей CLIPS

Цель: Изучение возможностей CLIPS

План:

1. Обзор CLIPS
2. Работа с CLIPS
3. Пример команд CLIPS
4. Синтаксис определений

Сейчас на рынке доступно не так уж много экспертных оболочек (инструментов, предназначенных для создания экспертных систем). Несмотря на то, что CLIPS распространяется бесплатно, он весьма успешно конкурирует даже с самыми известными коммерческими проектами.

Первоначально CLIPS использовал только методологию обработки данных посредством правил. CLIPS версии 5.0, вышедший весной 1991 г., ввел в CLIPS две новые парадигмы программирования: процедурное программирование (подобное используемому в языках C или Ada) и объектно-ориентированное программирование (похожее на языки Common Lisp Object System — CLOS или Smalltalk). Объектно-ориентированный язык программирования, предоставляемый системой CLIPS, называется CLIPS Object-Oriented Language (COOL). Версия CLIPS 5.1, вышедшая осенью 1991 г., улучшала поддержку разработок с использованием новых возможностей CLIPS и усовершенствовала интерфейс CLIPS для X Window, MS-DOS и Macintosh. Версия CLIPS 6.0, вышедшая 1993 г., предоставляла поддержку разработки модульных программ и тесную интеграцию объектно-ориентированных возможностей CLIPS и возможностей, базирующихся на правилах.

Так как CLIPS обладал портативностью, расширяемостью, мощностью и низкой стоимостью, он получил широкое распространение в государственных организациях, индустрии и учебных заведениях. Разработка CLIPS помогла усовершенствовать возможности технологии производства экспертных

систем среди широкого диапазона приложений. Система CLIPS используется большим числом организаций, включая все отделения NASA, военные ведомства США, множество федеральных, правительственных и государственных организаций, университеты и большое число частных компаний.

CLIPS версии 6.1 был выпущен летом 1998 г. Очередная версия содержала несколько существенных улучшений. Во-первых, исходный код CLIPS стал совместим с C++. Теперь для его компиляции можно использовать любой ANSI C- или C++-компилятор. Во-вторых, в систему были добавлены несколько новых команд, предоставляющие возможность профилирования по времени выполнения конструкторов языка или определенных пользователем функций.

Последняя, доступная сейчас версия CLIPS 6.2, вышла в свет 31 марта 2002 г. Основными отличиями новой версии CLIPS являются поддержка разработки встроенных приложений, использующих CLIPS, и улучшенный интерфейс для Windows-версии, оптимизированный для использования на платформах Windows 95/98/NT.

Благодаря тому, что CLIPS является свободно распространяемым программным продуктом с доступными исходными кодами, в последнее время было выпущено множество программ и библиотек, усовершенствующих и дополняющих возможности CLIPS. Некоторые из этих продуктов являются собственностью выпустивших их компаний и предназначены для внутреннего использования или коммерческого распространения, другие, как и сам CLIPS, распространяются свободно. В качестве самых известных примеров подобных проектов можно привести DLL/OCX-библиотеку, позволяющую использовать механизм логического вывода CLIPS в ваших приложениях, FuzzyCLIPS, CLIPS++, CLIPS code generator.

Работа с CLIPS

Для демонстрации примеров, используемых в этой книге, будет применяться Windows-версия CLIPS 6.2. Несмотря на полную совместимость

с Apple Macintosh и UNIX-версиями, при работе с данной книгой желательно использовать именно Windows-версию среды CLIPS.

Windows-версия среды CLIPS полностью совместима с базовой спецификацией языка. Ввод команд осуществляется непосредственно в главное окно CLIPS. Однако по сравнению с базовой Windows-версия предоставляет множество дополнительных визуальных инструментов, значительно облегчающих жизнь разработчика экспертных систем.

Экспертные системы, созданные с помощью CLIPS, могут быть запущены тремя основными способами:

- вводом соответствующих команд и конструкторов языка непосредственно в среду CLIPS;
- использованием интерактивного оконного интерфейса CLIPS (например для версий Windows или Macintosh);
- с помощью программ-оболочек, реализующих свой интерфейс общения с пользователем и использующих механизмы знаний и логического вывода CLIPS.

Кроме того, CLIPS при запуске позволяет выполнять командные файлы собственного формата (эта возможность также доступна при помощи команды `batch`). Для реализации этой возможности необходимо запустить CLIPS (в нашем случае это файл `CLIPSWin.exe`) с одним из трех следующих аргументов:

- `-f <имя-файла>`
- `-f2 <имя-файла>`
- `-l <имя-файла>`

Текстовый файл, заданный с помощью опции `-f`, должен содержать команды CLIPS. Если заданный файл содержит команду `exit`, то CLIPS завершит свою работу и пользователь вернется в операционную систему. В случае если команда `exit` отсутствует, то после выполнения всех команд из заданного файла пользователь попадет в главное окно CLIPS. Команды в текстовом файле должны быть набраны так же, как если бы они вводились

непосредственно в командную строку CLIPS (т. е. все команды должны быть заключены в круглые скобки и разделены символом перехода на новую строку). Опция -f фактически эквивалентна запуску команды batch сразу после запуска CLIPS.

Опция -f2 идентична -f, но, в отличие от опции -f, она использует команду batch*. Файл, заданный этой опцией, также выполняется после запуска CLIPS, но результаты выполнения команд не отображаются на экране.

Опция -l задает текстовый файл, содержащий конструкторы CLIPS, которые тут же запускаются на выполнение. Использование этой опции эквивалентно использованию команды load сразу после запуска CLIPS.

Создание программ-оболочек, использующих возможности CLIPS, выходит за рамки этой книги. Желая использовать эту возможность CLIPS можно рекомендовать обратиться к книге "CLIPS Reference Manual, Volume II, Advanced Programming Guide".

Основным методом общения с CLIPS, используемым в данной книге, является применение командной строки. После появления в главном окне CLIPS приглашения — CLIPS > — команды пользователя могут вводиться в среду непосредственно с клавиатуры. Команды могут быть вызовами системных или пользовательских функций, конструкторами различных данных CLIPS и т. д. В случае вызова пользователем некоторой функции, она немедленно выполняется, и результат ее работы отображается пользователю. Для вызова функций или операций CLIPS использует префиксную нотацию — аргументы всегда следуют после имени функции или операции. При вызове конструкторов CLIPS создает новый объект соответствующего типа, так или иначе представляющий некоторые знания в системе. При вводе в среду имени созданной ранее глобальной переменной CLIPS отобразит ее текущее значение. Ввод в среду некоторой константы просто приведет к ее немедленному отображению в главном окне CLIPS.

Листинг 1. Пример команд CLIPS

(+ 3 4)

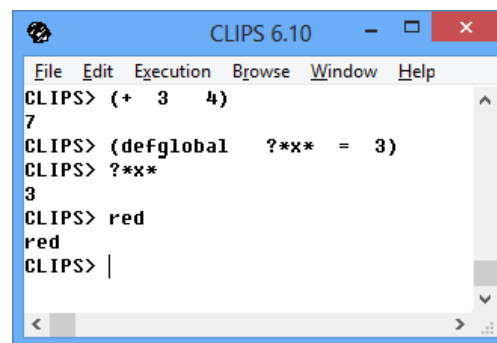
```
(defglobal ?*x* = 3)
```

```
?*x*
```

```
red
```

Результат выполнения этих команд приведен на рис. 22.

Первой командой данного примера вызывается функция арифметического сложения двух чисел: 3 и 4. Результат работы этой функции — 7 сразу же отображается на экран. После этого, с помощью конструктора `defglobal`, создается глобальная переменная `?*x*`, которая инициализируется значением 3. При вводе в командную строку имени глобальной переменной `?*x*` CLIPS отображает ее текущее значение, равное 3. Последней командой была введена некая константа `red`, значение которой было тут же выведено на экран.



```
CLIPS 6.10
File Edit Execution Browse Window Help
CLIPS> (+ 3 4)
7
CLIPS> (defglobal ?*x* = 3)
CLIPS> ?*x*
3
CLIPS> red
red
CLIPS> |
```

Рис. 22. Результат выполнения команд листинга 1

В заключение данной главы рассмотрим синтаксис, используемый в этой книге для определения тех или иных конструкций языка.

Синтаксис определений

В качестве базового синтаксиса для определения конструкций языка используется стандартная БНФ-нотация. Ниже приведены правила, используемые для построения определений.

Слово или выражение, заключенное в угловые скобки, называется нетерминальным символом (например, `<string>`). Нетерминальный символ требует дальнейшего определения. Слова или выражения, не заключенные в угловые скобки, называются терминальными символами, и представляют

синтаксис описываемой конструкции языка CLIPS. Терминальные символы (особенно круглые скобки) должны вводиться в командную строку именно так, как показано в определении. Если за нетерминальным символом следует символ *, то это означает, что в данном месте может находиться список из нуля или более элементов этого типа. Если же за нетерминальным символом следует +, то в данном месте может находиться список из одного или более элементов этого типа. Символы * и +, встречающиеся сами по себе (не следующие после нетерминальных символов), являются терминальными. Многоточие, как горизонтальное, так и вертикальное, также используется для отображения списка из одного или более элементов. Элементы, заключенные в квадратные скобки (например, [<комментарии>]), являются необязательными элементами, которые могут входить в определение. Вертикальная черта, разделяющая два или более элемента определения, указывает на то, что в конструкции необходимо использовать один из перечисленных элементов. Символ ::= используется для обозначения необходимости замены некоторого нетерминального символа. Например, определение:

$$\langle \text{lexeme} \rangle ::= \langle \text{symbol} \rangle | \langle \text{string} \rangle$$

обозначает, что нетерминальный символ <lexeme>, встречающийся в некотором определении, должен быть заменен либо на символ <symbol>, либо на символ <string>. Пробелы, символы табуляции, переходы на другую строку используются только для логического разделения элементов определения и игнорируются CLIPS (кроме строк, заключенных в двойные кавычки).

Контрольные вопросы:

1. Предназначение среды CLIPS.
2. Способы запуска ЭС в CLIPS.
3. Какая нотация используется в качестве базового синтаксиса для определения конструкций языка CLIPS?

Литература:

1. А. П. Частиков Т. А. Гаврилова Д. Л.Белов. РАЗРАБОТКА
ЭКСПЕРТНЫХ СИСТЕМ. СРЕДА CLIPS. СПб: БХВ-Петербург,
2003

Ключевые слова:

CLIPS, экспертные оболочки, примеры команд CLIPS.