OSTIS-2011
(Open Semantic Technologies for Intelligent Systems)

УДК 004.822

ПРИНЦИПЫ ОРГАНИЗАЦИИ ИНСТРУМЕНТАЛЬНОЙ СРЕДЫ И РАЗРАБОТКИ В НЕЙ ИНТЕЛЛЕКТУАЛЬНЫХ СИСТЕМ НА СЕМАНТИЧЕСКИХ СЕТЯХ (НА ПРИМЕРЕ IDE MULTI STUDIO)

В.А. Катаев (*bravo666666@yandex.ru*)
ОАО Пермская научно-производственная приборостроительная компания,
г.Пермь, Российская Федерация

Доклад содержит некоторые базовые принципы организации универсальной интеллектуальной метасреды для создания прикладных интеллектуальных систем на семантических сетях, возможные пути реализации этих принципов и демонстрацию некоторых из них на примере разрабатываемой метасистемы Multi Studio.

Ключевые слова: инструментальная среда, интеллектуальная система, алгоритмический язык, семантическая сеть.

Введение

В мире наблюдается стремительный рост вычислительных и коммуникационных мощностей при снижающемся КПД их использования. Для примера: по данным заместителя директора НИВЦ МГУ член-корреспондента РАН Владимира Воеводина мощности дорогостоящих суперкомпьютеров России используются всего на 5-6 процентов, да и то преимущественно в экспериментальном режиме. Домашние ПК у населения и того меньше. В основном это Интернет, игры и подготовка текстов.

По нашему мнению, основная причина - низкое качество программного обеспечения, индикаторами которого являются:

Низкий уровень алгоритмических языков, являющихся интерфейсом между человеком и компьютером и их излишнее разнообразие

Низкий уровень интеллекта информационных структур и их излишнее многообразие

Сложность программного обеспечения.

А также - сложность и многообразие физических устройств информационных систем, их несогласованность с виртуальными информационными структурами и между собой.

Для решения проблемы требуется соответственно:

Создание простого универсального интеллектуального языка связи человека с компьютером и компьютеров между собой.

Унификация виртуальных информационных структур на семантических сетях, как наиболее универсальной формы представления информации, являющейся сегодня наиболее близким аналогом человеческого мозга.

Создание физических устройств, обеспечивающих эффективную реализацию указанного языка и виртуальных семантических сетей.

Таким образом, главным в разработке информационных систем, повышения их интеллекта является уровень интеллекта человеко-машинного языка, лингвистическое программирование.

Информационные системы (интеллектуальные и не только) можно разделить на 2 класса:

Пользовательские информационные системы (ПИС)

Инструментальные среды создания ПИС – Мета информационные системы (МИС)

В разделе 1 рассматривается раздельный («компиляционный») вариант разработки ПИС в среде МИС и эксплуатации их за пределами МИС.

В разделе 2 рассматривается совмещенный («интерпретационный») вариант разработки и эксплуатации ПИС в среде МИС.

В разделе 3 в качестве примера реализации принципов совмещенного варианта представлена информация о системе Multi Studio на базе универсального языка Multi.

В разделе 4 рассматриваются сравнительные достоинства обоих вариантов, отмечаются перспективные преимущества второго и возможные пути реализации совмещенного варианта.

В разделе 5 подводятся общие итоги указанного сравнения.

1. Компиляционный вариант разработки и эксплуатации ПИС

Раздельная схема разработки и эксплуатации ПИС представлена в приложении на рисунке 1.

В левой части схемы находится ветвь разработки программ ПИС, а в правой – ветвь эксплуатации этих программ.

1.1. Разработка программы

Проектант (постановщик) задачи оформляет проект (алгоритм) задачи на языке ПРОЕКТО, являющимся некоторым «информационным» подмножеством своего национального диалекта естественного языка, которому его научили в образовательном учреждении. Проект (техническое задание) передается прикладному программисту, который разрабатывает программу решения задачи на некотором алгоритмическом языке АЛГО (C#, Java …) и производит отладку ее в соответствующей среде разработки программ (МИС, например MS Visual Studio).

Результатом является готовая к эксплуатации программа в кодах соответствующего вычислительного устройства (на языке ТЕХНО). Готовая программа системой МИС помещается в файловую систему ОС и там сохраняется.

В типовом случае программа состоит из 4 виртуальных модулей (ввод данных, их обработка, сохранение их в Базах знаний (БЗ) и вывод результатов). Данные в БЗ хранятся в форматах файловой системы на языках СУБД.

В разработке сложных ПИС принимают участие системные программисты.

1.2. Эксплуатация программы

Пользователь (специалист по эксплуатации) ПИС производит запуск программы на языке ОС, по запросу программы вводит исходные данные и получает результаты на языке Пользователя ПОЛЬЗО, являющегося, как и ПРОЕКТО, некоторым «информационным» подмножеством своего национального диалекта естественного языка, но менее профессионального, чем ПРОЕКТО.

2. Интерпретационный вариант разработки и эксплуатации ПИС

Совмещенная схема разработки и эксплуатации ПИС представлена в приложении на рисунке 2.

В левой части схемы по-прежнему находится ветвь разработки программ ПИС, а в правой – ветвь эксплуатации этих программ.

2.1. Разработка программы

Большую часть небольших проектов разрабатывают сами пользователи–непрограммисты на простом универсальном «семантическом» алгоритмическом языке Пользователя СЕМПО. В сложных случаях привлекаются Проектанты и прикладные Программисты.

Автор задания (программы) на СЕМПО запускает его в универсальной среде МИС, где Транслятор на основе текста задания формирует программу на «СЕМантическом КОмпьютерном» языке СЕМКО.

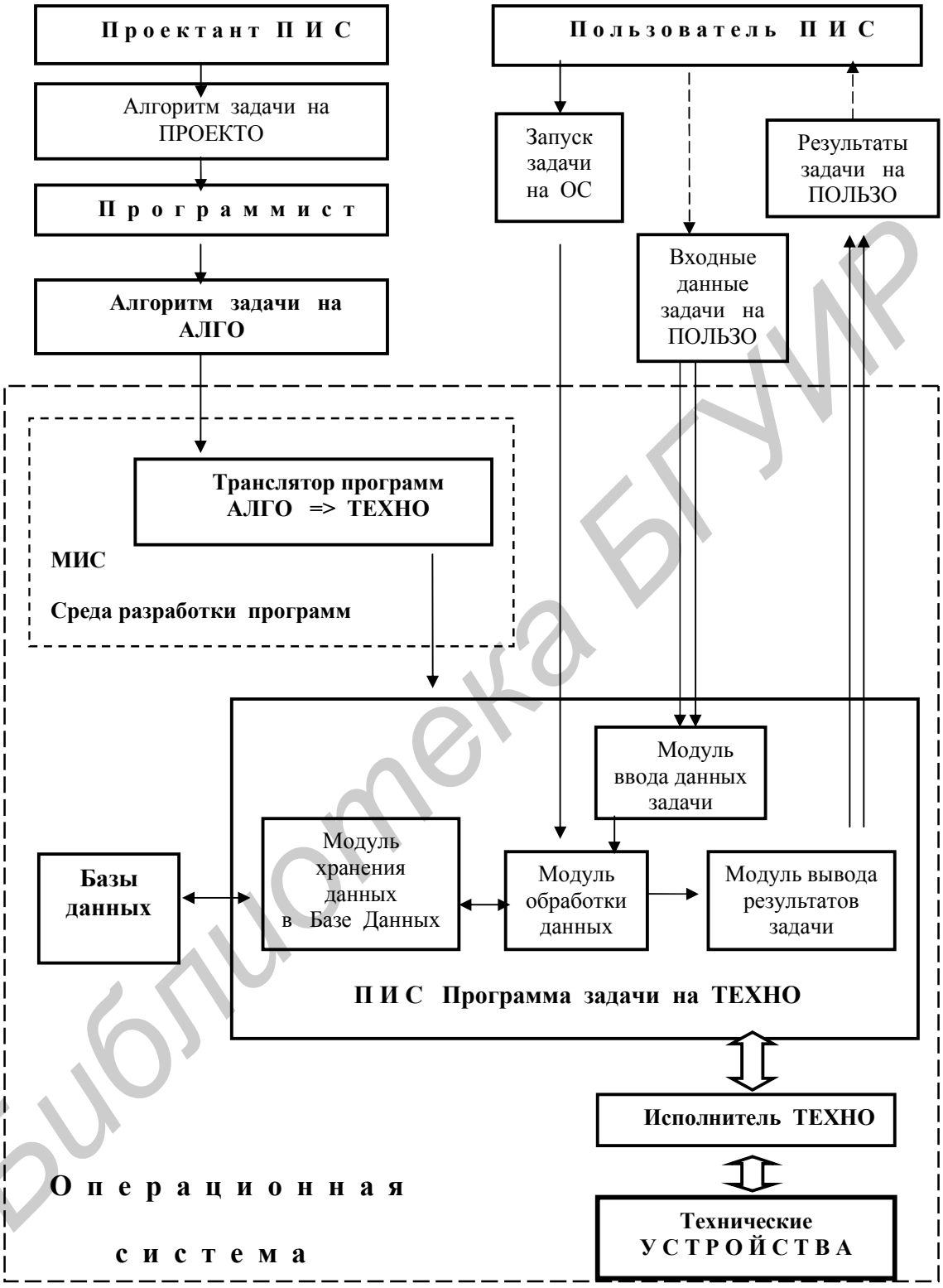

Рисунок 1 - Компиляционная схема разработки пользовательских программ

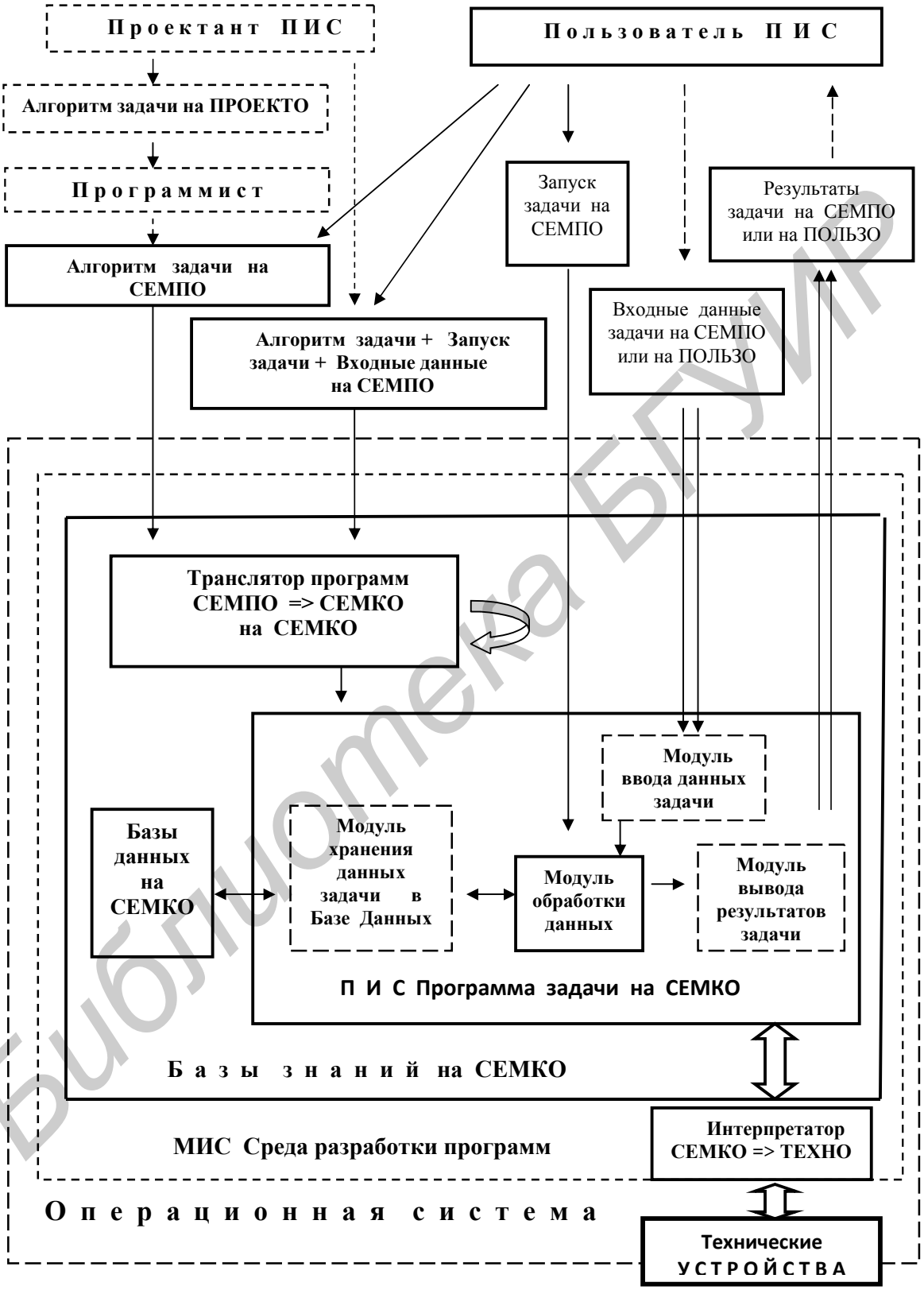

Рисунок 2 - Интерпретационная схема разработки пользовательских программ

В универсальной среде МИС все информационные объекты любой природы (знания, данные, программы их обработки) оформляются на этом языке в виде семантических сетей, в узлах которых располагаются базовые (терминальные) информационные объекты, а также отношения между ними. Сложные объекты состоят из набора более простых. Взаимные ссылки на именованные объекты создают виртуальные базы знаний динамического размера. Разрешены итерации, рекурсии, синонимы, омонимы, контексты и другие языковые конструкции, свойственные естественному языку.

СЕМПО – язык расширяемый, постоянно дополняемый все более мощными типовыми понятиями, свойственными человеческому мышлению. Имеет все основные парадигмы, присущие естественному языку (от функциональной до логической).

Взаимнооднозначное соответствие языков СЕМПО и СЕМКО позволяет производить обратную ретрансляцию программы СЕМКО => СЕМПО.

2.2. Эксплуатация программы

Готовая СЕМКО-программа исполняется немедленно и/или сохраняется в Базе Знаний для последующего использования, не покидая среды МИС.

Исполнение программы производится интерпретатором СЕМКО => ТЕХНО.

Унифицированная форма записи входных данных и программ позволяет Пользователю работать в диалоговом режиме и вводить их одновременно в одном задании.

Вывод на ПОЛЬЗО.

3. Реализация некоторых принципов универсального интерпретационного варианта в системе Multi Studio

3.1. Система Multi Studio

В Пермском Центре интеллектуальных технологий разрабатывается мета инструментальная система Multi Studio (MS).

Конечной целью разработки системы MS является создание программно-языкового инструментария для общения человека с интеллектуальным роботом. На данный момент система позиционируется как средство обучения школьников, студентов программированию и как инструмент для решения различных математических и прикладных задач.

Центральной частью системы является Бункер (База Знаний на семантических сетях).

В системе различаются два языка: язык среды (ПОЛЬЗО) и язык заданий (СЕМПО).

Язык среды – это национальный язык, фразы которого присутствуют на панелях системных окон и на котором система приветствует пользователя, сообщает о наличии ошибок и т.п. Базовым диалектом языка среды является русский. Имеется возможность простой пользовательской настройки на другие национальные диалекты.

Языком заданий является Multi. Базовые обозначения команд языка имеют смешанный диалект (частью символьные, частью аббревиатура национальных слов) и содержатся в словаре системы. Пользователь может дополнять базовый словарь своими синонимами команд или создать новый словарь. В MS программы (задания) обработки данных и сами данные имеют одинаковую унифицированную структуру. Синтаксический анализатор то и другое переводит на внутренний язык (СЕМКО) в графы, узлами которых становятся отдельные слова. При этом в графе задания будут преобладать команды, а в информационном графе – данные (литералы, числа, даты и др.).

Различают внешний и внутренний язык Multi.

Внешний язык – это текст на Multi, вводимый с клавиатуры (или с микрофона) и выводимый на внешнее устройство компьютера (на экран, на принтер, на аудиоколонки).

Внутренний язык Multi – это закодированные определенным образом и записанные в Бункер системы слова внешнего языка.

При вводе текста задания на диалекте Multi система переводит его из внешнего представления во внутренний код.

Система выполняет задание путем интерпретации его внутреннего кода. При этом поименованное задание может быть сохранено в Бункере полностью или частично для последующего воспроизведения путем динамического вызова по имени. Пользователь

может произвести обратный перевод из внутреннего представления в любой внешний диалект. Таким образом, программа может быть написана на русском диалекте, а прочтена на французском.

Далее примеры текстов языка представлены на некотором демонстрационном диалекте, близком к научному.

3.2. Базовая структура языка

Базовый алфавит Multi включает в себя все символы англо-русской клавиатуры. Для обозначения команд (функций) используются преимущественно классические математические обозначения, по возможности нейтральные по отношению к национальным языкам.

Задание на языке Мулти состоят из предложений, заканчивающихся точкой с запятой. Предложение состоит из списка выражений. Выражение состоит из слова и аргументного списка. Аргументный список (аргум) – это перечень аргументов, заключенных в круглые скобки. Аргум может отсутствовать. Аргументом языка может быть слово или выражение.

Глубина вложений выражений друг в друга зависит от реализации (сегодня 1000 уровней). Конечный символ предложения ***';'*** закрывает все не закрытые скобки ***')'*** аргумов. Слова языка отделяются друг от друга произвольным числом пробелов.

Слова могут быть командами и другими информационными объектами различных типов. Различаются базовые (абстрактные) и пользовательские типы данных. Базовые типы – это числа до 17 разрядов (***1988 +3.5***), тексты неопределенной длины (***«Петрова»***), имена (***Расчет_зарплаты***), даты (***2011.02.11***), мультимедийные файлы и другие.

Типы переменных не задаются в явном виде. Одна и та же переменная в процессе выполнения задания может быть модулем программы, базой данных, числом, текстом. Смысл объектов базовых типов определяется Пользователем по умолчанию.

Пользовательские типы явно задают смысл данного в формате значение:тип (***1988:год 3.5:кг «Петрова»:фамилия 09:30:00:начало_работы***).

Сейчас в Multi Studio реализовано свыше 500 команд (ввода-вывода, арифметики, логики, управления базами данных, генерации отчетов, экспертных систем и других). Их число постоянно пополняется по мере расширения предметной области системы.

Пример задания:

P# (12 + (77 Факториал) «привет» ; Факториал(*(..n (+n ;

Задание состоит из двух предложений и читается так:

Через пробел вывести на экран (***P#***) число (***12***), сумму (+) числа (***77***) и вычисленного значения переменной по имени (***Факториал***). Конец предложения (***;***)

Факториал есть произведение (*******) возрастающего ряда чисел (***..n***) начиная с единицы (по умолчанию) с шагом 1 (тоже по умолчанию) по число, которое следует ввести (***+n***) в открывающееся поле на экране системы. Затем вывести текст (***«привет»***). Конец предложения (***;***)»

При этом блок ***Факториал*** автоматически запоминается в Базе Знаний.

3.3. Некоторые парадигмы языка Multi

Multi имеет **функциональную** префиксную синтаксическую структуру выражений, но при этом у него присутствуют не только рекурсивные, но и итеративные циклы, а также команды присваивания.

Multi-программы легко структурируются с использованием именованных модулей, в качестве которых могут выступать любые части программы.

В Multi имеет место **императивно-декларативный метод** программирования.

В общем случае предложения исполняются в смешанном порядке: неименованные предложения выполняются в порядке их нахождения в тексте задания, именованные – только в порядке их вызова.

Multi имеет средства для эффективного **логического программирования**:

База знаний *Соседи* на русском языке:

Николай – 1960 года рождения. Он отец Нины и Ивана. Иван и Галя - родители Павла. Николай любит соседку, которую зовут Люба, а также любит охоту.

Дед – это отец родителя.

На Multi это можно записать так:

```
Соседи (   Николай ( 1960:год_рождения
                     отец (Нина Иван ( родитель (Павел) ) )
                     любит ( соседка (Люба) охота )   )
                Галя ( родитель (Павел) ) ) ;
^^дед ( ^отец ( ^родитель ));
```

Примеры запросов к Базе ***Соседи***:

Все данные о Николае: ***w. (Соседи (Николай (??)));***

Есть ли связь между Николаем и Любой: ***w. (Николай (??(Люба)));***

Кто является дедом у Павла: ***w. (? (дед (Павел)));***

В Multi присутствуют такие понятия **объектно-ориентированного программирования**, как ‘класс’ и ‘экземпляр класса’:

^отец : фамилия - классы

отец Попов:фамилия - экземпляры классов

Контекстное программирование:

В Multi широко представлена контекстная зависимость объектов языка друг от друга. Например, в области действия команды ***test.*** помимо основного вывода производится пошаговая трассировка командных переходов и вывод результатов выполнения каждой команды в заданное окно.

Наличие омонимов. Пример: ‘-‘ это команда вычитания, но если ‘-‘ аргумент команды ***?-+*** , он указывает, что при обнаружении ошибки следует прекратить выполнение задания.

Параллельное программирование:

A |||(B C D) E F;

Сначала выполняется модуль A. Затем параллельно в любой последовательности B, C и D. После их завершения выполняется E, а за ним модуль F.

Язык управления базами данных и базами знаний:

Для ввода, корректировки и поиска данных имеется группа команд низкого, среднего и высокого уровня.

w.(Соседи (+d (Соня (мать (Федор (студент имеет (авто; // +d вставить в БД

Язык разработки экспертных систем:

запросно-ответные команды типа:

```
vod. ( “годовая оценка ?»   1 («оставить на второй год!»)
                            2(«на осеннюю пересдачу» ) 3 4
                            5(«выдать похвальный лист»;
```

Multi – метаязык:

Фрагмент описания Multi на Multi:

Multi ([.] (предложение ([.] (выражение) “;” ;

Читается: ***“Multi – непустое множество предложений (каждое из которых есть непустое множество выражений) и заканчивается символом ;»***

3.4. Порядок функционирования системы Multi Studio

Система по заданиям Пользователя производит различные математические расчеты (арифметика, тригонометрические и другие функции), создает Базы данных сложной нерегулярной структуры, производит поиск в них, генерирует отчеты.

Диалог Пользователя с компьютером протекает в многооконной среде. Каждое окно имеет индивидуальную панель управления.

Два окна используются в качестве справочников по языку.

Третье окно используется для редактирования текста Multi на текущем диалекте. Тексты заданий могут вводиться с клавиатуры компьютера или считываться из файлов.

Синтаксические ошибки указываются цветом прямо на тексте с точностью до символа, а также комментируются в окне протокола.

Транслятор формирует задание во внутреннем коде во временной памяти и записывает поименованные программные модули и блоки данных в Бункер. Интерпретатор системы обходит узлы задания в заданной последовательности и исполняет находящиеся в узлах команды.

Поиск и корректировка данных в Базах производится на нижнем уровне путем обхода графов Баз произвольной нерегулярной структуры с использованием команд навигации. Среднеуровневые команды позволяют обходить Базы с использованием схем (шаблонов), описывающих структуру Баз. Высокоуровневые команды сами организуют навигацию по Базам, избавляя пользователя от этой рутины.

Вывод результатов выполнения задания может производиться параллельно в 5 системных окон и/или в файлы. Выходные данные можно перетаскивать из одного окна в любое другое и корректировать их перед выводом на принтер и/или в файл и/или вновь в систему.

Система позволяет протоколировать беседу пользователя с компьютером в окне протокола.

Управление памятью автоматическое.

Ввод заданий, процесс их выполнения и результаты вычислений могут озвучиваться сказочными персонажами (Ассистентами) на русском (а также английском) языке. Пользователь может масштабировать тексты в системных окнах. Это позволяет работать с системой пользователям с пониженным зрением.

Реализован интерфейс с известной математической системой MatLab.

Система работает в среде Windows и Linux (кроме некоторых функций, требующих доработки).

3.5. Экспериментальная проверка системы Multi Studio

В процессе разработки Multi Studio в 2007-2009 гг. производилось неоднократное контрольное тестирование системы путем определения усвояемости языка Multi школьниками начиная с третьих классов и студентами ВУЗов и сравнительным анализом Multi с классическими языками (Basic, Pascal, Prolog, начальный SQL).

Программы, написанные на Multi, как правило, в несколько раз короче написанных на других языках.

В частности, было произведено сравнение Multi Studio с известной американской инструментальной системой разработки экспертных систем CLIPS на примере американской же тестовой системы поиска неисправностей двигателя. Multi превзошел CLIPS по основным параметрам (простоте языка, длине программы, быстродействию, эргономике системы, мощности алфавита, по наличию в MS синтеза речи).

Общее время изучения Multi Studio на порядок короче, чем изучение классических алгоритмических языков с соответствующими средами подготовки программ и с аналогичной функциональностью.

3.6. Дальнейшее развитие системы Multi Studio

В ближней перспективе предполагается увеличить функциональность и уровень имеющихся групп команд , реализовать распределенные вычисления, управление знаниями, разработать дополнительный графический интерфейс с пользователем, информационные интерфейсы с некоторыми программными продуктами широкого применения.

Провести широкое опробование Multi Studio в нескольких предметных областях: в образовании (обучение информатике), в промышленной сфере (создание экспертных систем). А также в разработке других интеллектуальных систем.

4. Сравнение вариантов разработки и эксплуатации ПИС

4.1. Компиляционный вариант

Недостатки:

- Завышенное многообразие и низкий уровень алгоритмических языков
- Почти монопольное применение английского языка в качестве системного
- Сложности программирования параллельных вычислений
- Многообразие операционных систем, файловых систем и СУБД

Высокие требования к разработчикам ПИС и к эксплуатационному персоналу
Достоинства:
Многолетние наработки программного и технического обеспечения
Многомиллионная армия готовых специалистов
Независимость готовых программ от МИС
Высокое быстродействие исполнительных программ

4.2. Интерпретационный вариант (универсальный семантический)

Недостатки:

- Отсутствие опыта успешного массового применения эффективных универсальных языков класса СЕМПО
- Отсутствие комплексов технических устройств, ориентированных на поддержку языков класса СЕМКО
- Более низкая скорость работы интерпретаторов против скомпилированных программ

Достоинства:

- Простота языков СЕМПО, приближение их уровню естественных языков
- Диалоговый режим работы в среде МИС
- Эффективное управление параллельными вычислениями
- Резкое повышение производительности разработчиков ПИС и повышение надежности ПИС за счет высокого уровня команд языка.
- Резкое расширение пользовательского электората (от школьников до академиков), использующих компьютеры в различных областях в диалоговом режиме
- Сокращение числа Проектантов и прикладных Программистов, занятых разработкой ПИС
- Высокий интеллект ПИС за счет высоты языка СЕМПО и работы СЕМКО - программ с унифицированными Базами Знаний

О быстродействии

- Безусловно, операторы одного уровня скомпилированных программ работают быстрее, чем интерпретатор обрабатывает эти же операторы.
- Однако СЕМКО позволяет иметь команды более высокого уровня, которые производят сразу несколько операций, да еще и в циклах.
- Кроме того обработка данных в разнородной многофайловой системе требует значительной работы по извлечению, реструктуризации, перекодированию и перемещению данных в отличие от аналогичной работы в семантической сети.
- Экспериментальная практика применения системы Multi Studio скорее подтверждает вышесказанное.
- Использование параллельных вычислений позволяет даже превзойти скомпилированные программы по интегральному быстродейсвию.
- Наконец, стремительно растущая производительность новых суперкомпьютеров вообще снимает проблемы сравнительного быстродействия.

Об объеме программ и данных

- С одной стороны одноуровневые операторы скомпилированных программ требуют меньше памяти, чем аналогичные команды СЕМКО, занимающие универсальные элементы семантических сетей.

- Однако сохранение программы в родительской среде МИС позволяет ей во многих случаях «бесплатно» пользоваться типовыми сервисными средствами среды (ввод данных и вывод результатов в формате СЕМПО, а также хранение данных в Базе знаний).
- То - есть наличие этих модулей в СЕМКО-программе во многих случаях необязательно (что отражено в схеме пунктирной окантовкой соответствующих блоков).
- В принципе то же можно сказать о данных.
- Нерегулярная структура баз данных в семантической сети позволяет не занимать пустого пространства в памяти для отсутствующих показателей.
- Кроме того общий пул свободной памяти в базах знаний позволяет не создавать резервной памяти для отдельных программ и баз данных.

Об операционных системах и МИС

- В интерпретационном варианте наличие объемлющей операционной системы кажется излишним, так как все основные вычислительные процессы проходят в МИС. Здесь же идет управление Базами знаний.
- При частичном использовании МИС для узких предметных областей МИС может поставляться Пользователям с сокращенной фукциональностью (ядро МИС + необходимые функциональные модули), что сокращает потребность в памяти, а также увеличивает быстродействие.

Заключение

Общие выводы:

Универсальный интерпретационный вариант разработки и эксплуатации ПИС в перспективе имеет несомненные преимущества перед классическим компиляционным вариантом.

Однако массовый переход на новую технологию будет постепенным. И вероятно займет от 7 до 15 лет.

Новая технология позволит создать Международную Сеть Баз Знаний и на ее основе Семантический Веб.

При этом, несомненно, будут всегда сосуществовать конкурирующие локальные технологии, как первого варианта (например, для экстремальных космических вычислений), так и второго варианта (типа MatLab).

Применение универсальных языков класса СЕМАПО/СЕМАКО и соответствующих информационных технологий в международном масштабе создаст предпосылки для разработки эффективного международного языка связи, являющегося расширением СЕМАПО в сторону простого искусственно-естественного языка типа эсперанто.

В связи с этим представляется, что нынешняя разработка аналогичного языка UNL, которая проводится с 1995 года под эгидой ООН, является несколько преждевременной, поскольку базируется на упрощенном (но, тем не менее, сложном) английском языке и на действующих разномастных информационных и компьютерно - лингвистических технологиях нескольких ведущих стран мира.

Библиографический список

[Голенков и др., 2001] Представление и обработка знаний в графодинамических ассоциативных машинах. В.В.Голенков, О.Е.Елисеева, В.П.Иващенко и др.: Под редакцией И.И.Голенкова. – Мн.:БГИУР,2001.

[Катаев, 2008a] Катаев В.А. Компьютерный эсперанто для искусственного интеллекта. //Философскометодологические проблемы искусственного интеллекта. // Материалы Всероссийского междисциплинарного семинара 1-2 ноября 2007 г. –Пермь: ПГТУ, 2008.

[Катаев, 2008b] Катаев В.А. Разработка диалоговой среды Multi Studio для общения с компьютером на языке нового поколения. // Перспективы развития телекоммуникационных систем и информационные технологии. Труды международной конференции – СпБ.: Издательство Политехнического университета, 2008.

[Катаев и др., 2009] Катаев В.А., Еремин Е.А. Непрерывное электронное обучение населения информатике с младшего возраста до преклонных лет на базе диалоговой системы Multi Studio. // Материалы международной конференции 6-8 октября 2009 г. «Электронная культура. Информационные технологии будущего и современное электронное обучение «MODERN IT & (E-) LEARNING». – Астрахань: ООО «Типография «НОВА», 2009.