СПЛАЙН-ФУНКЦИИ В ИНЖЕНЕРНОЙ ГЕОМЕТРИИ
(Вычислительные системы)
1981 год Выпуск 86

УДК 519.682

СИСТЕМА АВТОМАТИЗАЦИИ ГЕОМЕТРИЧЕСКИХ РАСЧЕТОВ И ЕЕ ВХОДНОЙ ЯЗЫК

В.М. Гамидов

Автоматизация проектно-конструкторских работ и технологической подготовки производства средствами вычислительной техники требует создания: I) математического аппарата для представления и обработки геометрической информации в ЭВМ; 2) алгоритмов и программных средств построения математической модели изделия; 3) средств для осуществления разнообразных операций на математической модели.

Системное использование этих средств требует рассмотрения перечисленных задач во взаимосвязи друг с другом, с учетом способа решения каждой и технических условий функционирования. При этом необходимо решить вопросы создания системных программных средств, позволяющих в своих рамках на единой основе увязать вторую и третью задачи, структуры данных, приемлемой для каждой задачи, и удобных для широкого круга пользователей средств общения.

Анализ существующих методов и программных средств, решающих перечисленные задачи, и опыт их применения выявили необходимость и возможность обособления геометрической информации, автономность и долгосрочность ее хранения, поскольку геометрическая информация используется при решении любых задач и имеются математические и технические средства ее получения и хранения. Выбор структуры данных должен учитывать этот факт, а язык общения - отражать эту возможность.

В работе [I] была сделана попытка создания языковых средств, позволяющих раздельно задавать геометрическую и технологическую информации. В дальнейшем этот подход получил развитие и привел к созданию и реализации системы автоматизированного расчета и изготовления сложных конструкций (РИСК) [2] в рамках пакетного режима

работы ДОС ЕС ЭВМ.Система РИСК является в определенном смысле инструментальной системой программирования для решения второй и третьей задач,в основу ее положен математический аппарат сплайн-функций [3]. Данная статья имеет целью ознакомление с системой РИСК, ее структурой,входным языком и перспективами дальнейшего развития.

I. Общее описание системы

Схематично структура математического обеспечения системы РИСК представлена на рис. I.

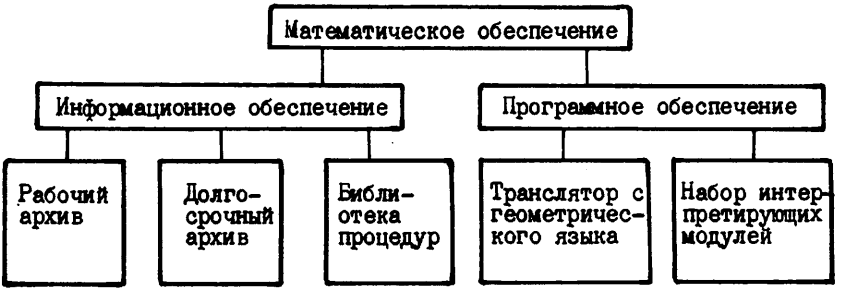

Рис. I

Любой объект в системе характеризуется пятью параметрами: K, L, M, P, Q . Параметры K, L, M задают трехмерную структуризацию данных,определяющих объект (такая структуризация используется интерпретирующими модулями); P - характеризует тип объекта; Q - определяет подтип данного типа. Параметры P и Q используются на языковом уровне и уровне интерпретирующих модулей.

Такой выбор внутренней структуры объектов в системе органически увязан с математическим аппаратом представления геометрических объектов. В качестве иллюстрации приведем некоторые геометрические объекты и их возможную внутреннюю структуризацию по параметрам K, L, M.

Геометрический объект "точка" может быть определен: I) как "точка в пространстве" заданием координат X, Y, Z, тогда структуризация имеет вид: $K=3, L=1, M=1$; 2) как"точка на кривой" заданием координат X, Y, Z на кривой и характеристики самой кривой в окрестности этой точки,например, касательного вектора к кривой в данной точке. т.е. заданием X', Y', Z', тогда $K=6, L=1, M=1, P=T$.

Геометрический объект "поверхность" определяется набором кар-

касных линий, каждый узел каркаса задается четырнадцатью числами, определяющими координату узловой точки, вектор касательной по каждому направлению каркасных линий, значения их параметров в точке, вектор смешанной производной. Тогда структуризация имеет вид: $K = 14$, $L = \ell$, $M = m$, $P = \Pi$, где ℓ определяет количество точек в каркасной линии, а m указывает число таких каркасных линий по другому направлению.

Для расчета конкретных геометрических объектов имеется набор подпрограмм, из которых компонуется программа, интерпретирующая определенную геометрическую конструкцию входного языка. В дальнейшем такие программы называются геометрическими модулями. Результатом работы геометрических модулей является математическая модель геометрического объекта. Математические модели геометрических объектов на время работы системы хранятся в рабочем архиве системы.

Общение геометрических модулей с рабочим архивом осуществляется средствами управления архивами. Этими средствами реализуются следующие функции: опрос структуры объекта, чтение части объекта по измерениям K,L и K,L,M (L и M изменяются от I до верхнего предела), запись объекта в архив, запись части объекта к предыдущей части, копирование объектов из архивов и другие. Для долговременного хранения информации предусмотрен долгосрочный архив. Логические структуры архивов одинаковы.

Библиотека процедур системы предназначена для хранения программ, написанных на входном языке и оформленных в виде процедур.

Транслятор с геометрического языка осуществляет общее управление вычислительным процессом в системе и трансляцию программы. Он обладает широкими диагностическими возможностями.

2. Основные символы языка

Язык строится из следующих групп основных символов:

⟨основной символ⟩ :: = ⟨цифра⟩|⟨ буква⟩|⟨ знак операции⟩|⟨разделитель⟩|⟨ спецификатор⟩|⟨ служебное слово ⟩|⟨описатель ⟩

⟨цифра ⟩:: = 0|I|2|3|4|5|6|7|8|9

⟨буква⟩ :: = A|B|C|Д|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|Z|Y|X|Б|Г|Д|Ж|З|И|Й|Л|П|У|Ф|Ц|Ч|Ш|Щ|Ь|Ы|Э|Ю|Я

⟨знак операции ⟩:: = ⟨ арифметическая операция ⟩|⟨ тип геометрического оператора ⟩

⟨арифметическая операция ⟩:: = +|-|*|/|↑

Арифметические операции применяются для образования арифметических выражений.

⟨тип геометрического оператора⟩ :: = @ | =

Символ @ обозначает знак эквивалентности, используется в операторе эквивалентности, а также является указателем шифра.

Равенство используется в операторе присваивания.

⟨разделитель⟩ :: = (|) | ␣ | . | E | , | ; | : | #

Скобки (и) применяются в различных конструкциях языка для тех же целей, для каких они применяются в обычном математическом языке. В скобки заключаются параметры функций и процедур, арифметические выражения, которые нужно рассматривать как единое целое, а также конструкции геометрического выражения. Символ ␣ употребляется для построения строк входной программы и удоства ее зрительного восприятия. Десятичная точка и основание степени E применяются для построения чисел. Запятая служит для разделения параметров, относящихся к одному классу объектов. Точка с запятой употребляется для отделения определений понятий и ссылок на них. Двоеточие служит для отделения списка целых чисел в конструкциях и списка параметров типа VALUE от спецификатора в описании процедуры. Знак # служит для обозначения конца оператора и конца строки, если оператор занимает несколько строк.

⟨спецификатор⟩ :: = VALUE

Используется при спецификации параметров в описании процедуры.

⟨служебное слово⟩ :: = ⟨основное служебное слово⟩ | ⟨модификатор⟩

⟨основное служебное слово⟩ :: = ⟨тип геометрического объекта⟩ | ЦЕЛОЕ

⟨тип геометрического объекта⟩ :: = ТОЧКА | ВЕКТ | ПЛОСК | КРИВ | ПОВЕР | ГРАН

Служебные слова употребляются при построении операторов входного языка системы. Основные служебные слова, кроме ЦЕЛОЕ, используются при описании неарифметических выражений и могут стоять в геометрических операторах только перед скобкой. Слова этой группы задают тип основных объектов системы. Служебное слово ЦЕЛОЕ используется при описании процедуры и задает тип параметра процедуры.

⟨модификатор⟩ :: = ⟨характер типа⟩ | ⟨селектор⟩ | ⟨ ввод-вывод ⟩

⟨характер типа⟩ :: = ЭКВ | ЛИН | ЛЕНТ

Характером типа определяется подтип геометрического объекта: ЭКВ - "эквидистантная", ЛИН - "линейчатая", ЛЕНТ - "ленточная" поверхности.

⟨ селектор ⟩:: = КАС|ПЕР|ПАР|ХБ|ХМ|УБ|УМ| ZБ|ZМ

Слова этой группы позволяют делать выбор среди нескольких объектов.

⟨ ввод-вывод ⟩ :: = ВК|ВЛ|МЛ

Слова группы "ввода-вывода" указывают, что информация об определяемом геометрическом объекте должна быть введена с внешних устройств соответственно с перфокарт, перфоленты или магнитной ленты. В операторах после этих слов через запятую должен стоять номер или идентификатор, указывающий на внешнее устройство.

⟨ описатель ⟩:: = СТАРТ|ФИНИШ|КОММТ|ПРОЦЕД|КОНЕЦП

СТАРТ описывает начало программы; ФИНИШ указывает, что программа закончена; ПРОЦЕД указывает на начало описания процедуры; КОНЕЦП указывает на конец описания процедуры; после слова КОММТ можно писать любой текст примечания, признаком конца которого будет возврат каретки с переводом строки.

3. Основные конструкции языка

Основными конструкциями языка являются: числа, идентификаторы, функции.

⟨ вещественное число ⟩:: = ⟨ вещественное без показателя ⟩| ⟨ вещественное без показателя ⟩ Е ⟨ знак ⟩⟨ целое ⟩

⟨ вещественное без показателя ⟩:: = ⟨ целое ⟩|. ⟨ целое⟩|⟨ целое ⟩. ⟨ целое⟩

⟨ целое ⟩:: = ⟨ цифра ⟩|⟨ целое⟩ ⟨ цифра ⟩

⟨ знак ⟩:: = +| -

Обозначая буквой m целую часть числа, буквой n - дробную, буквой K - показатель степени, можно записать вещественное число в виде $m.n$ либо $m.nE\pm K$. Целая или дробная части числа могут отсутствовать.

Диапазон представления чисел порядка $|10^{76}| - |10^{-78}|$.

⟨ идентификатор⟩ :: = ⟨ буква ⟩|⟨ идентификатор ⟩⟨ буква ⟩| ⟨идентификатор ⟩⟨ цифра⟩

Идентификаторы используются для обозначения (наименования) переменных, стандартных, скалярных функций, процедур. Длина идентификаторов ограничена шестью символами.

З н а ч е н и е - это упорядоченное множество чисел (в частности, одно число или один вектор и т.д.). П е р е м е н н а я - это величина, значение которой вычисляется и может изменяться во время выполнения программы. В языке различают п р о с т ы е

переменные (они иногда будут называться а р и ф м е т и ч е с к и м и) и г е о м е т р и ч е с к и е переменные. Простые переменные - переменные, которым присваивается числовое значение. Геометрические переменные - переменные, идентификаторы которых обозначают геометрический объект. Различие между этими переменными определяется оператором присваивания.

⟨ функция ⟩:: =⟨ функция стандартная ⟩| ⟨ функция скалярная ⟩

Функция определяет одно числовое значение.

⟨ функция стандартная ⟩:: =⟨ идентификатор стандартной функции ⟩|⟨ арифметическое выражение ⟩|

⟨ идентификатор стандартной функции ⟩:: = ABS|SIGN|SIN|COS|SQRT|TAN|ARSIN|ARCOS|LN|LG|ARTAN|EXP|ENT

Приведенные стандартные функции имеют следующий смысл: ABS(A) - абсолютное значение арифметического выражения A; SIGN(A) - знак значения A; SIN(A) - синус A; A - задается в радианах; COS(A) - косинус A; SQRT(A) - корень квадратный из A, $\sqrt{A}$, $A \geq 0$; TAN(A) - тангенс A; ARSIN(A) - главное значение синуса A в радианах ; $|A| \leq 1$; ARCOS(A) - главное значение косинуса A; ARTAN(A) - главное значение тангенса; LN(A) - натуральный логарифм A , $A > 0$; LG(A) - десятичный логарифм A; $A > 0$; EXP(A) - экспоненциальная функция A; ENT(A) - целая часть A.

⟨ функция скалярная ⟩:: =⟨ идентификатор скалярной функции ⟩ (⟨список ссылок ⟩)

⟨ идентификатор скалярной функции ⟩:: = ANGLE|DIST|SCAL

⟨ список ссылок ⟩:: =⟨ идентификатор ⟩|⟨ список ссылок ⟩; ⟨ идентификатор ⟩

Эта группа функций ставит в соответствие нескольким геометрическим объектам число. Список идентификаторов скалярных функций может быть расширен, а сами функции определяют ANGLE(A;B) - величину угла между векторами A и B, DIST(A;B) - расстояние между объектами A и B, SCAL(A;B) - скалярное произведение векторов A и B .

4. Вычисление и присваивание значений

Арифметическое выражение задает порядок выполнения операций для получения числового значения. Эти операции производятся над фактическими числовыми значениями первичных выражений. При вычислении значения арифметического выражения принят обычный порядок выполнения арифметических операций.

⟨ арифметическое выражение ⟩ :: = ⟨терм ⟩ | ⟨ знак ⟩ ⟨ терм ⟩ | ⟨арифметическое выражение ⟩ ⟨ знак ⟩ ⟨ терм ⟩

⟨ знак ⟩ :: = + | -

Понятию терма соответствует понятие "одночлен без знака".

⟨ терм ⟩ :: = ⟨ множитель ⟩ | ⟨ терм ⟩ * ⟨ множитель ⟩ | ⟨ терм ⟩ / ⟨ множитель ⟩

Множитель представляет собой одно или несколько первичных выражений, разделенных знаками возведения в степень. Показатель степени не заключается в скобки только тогда, когда он является числом без знака, переменной, функцией стандартной или функцией скалярной. При многократном повторении операции возведения в степень эти операции выполняются последовательно в порядке их написания.

⟨ множитель ⟩ :: = ⟨ первичное выражение ⟩ | ⟨ множитель ⟩ ↑ ⟨ первичное выражение ⟩

⟨ первичное выражение ⟩ :: = ⟨ вещественное число ⟩ | ⟨ идентификатор ⟩ | ⟨ функция стандартная ⟩ | ⟨ функция скалярная ⟩ | (⟨ арифметическое выражение ⟩)

Геометрическое выражение задает способ получения геометрического объекта и ставит в соответствие набору заданных геометрических объектов и арифметических выражений некоторый геометрический объект, тип которого определяется в задании. В результате вычисления геометрического выражения получается набор числовых значений, задающих описываемый объект.

⟨ геометрическое выражение ⟩ :: = ⟨ тип геометрического объекта ⟩ (⟨ конструкция ⟩)

Тип геометрического объекта определяется подмножеством основных служебных слов.

⟨ тип геометрического объекта ⟩ :: = ТОЧКА | ВЕКТ | ПЛОСК | КРИВ | ПОВЕР | ГРАН

⟨ конструкция ⟩ :: = ⟨ характер типа ⟩ ; ⟨ список параметров ⟩ | ⟨ селектор ⟩ ; ⟨ список параметров ⟩ | ⟨ ввод-вывод ⟩ , ⟨ номер устройства в/в ⟩ ; ⟨ список параметров ⟩ | ⟨ список параметров ⟩

⟨ список параметров ⟩ ::= ⟨ числовые данные ⟩ | ⟨ список ссылок ⟩ | ⟨ числовые данные ⟩ ; ⟨ список ссылок ⟩ | ⟨ список ссылок ⟩ : ⟨ список целых ⟩ | ⟨ список ссылок ⟩ ; ⟨ числовые данные ⟩ | ⟨ список значений ⟩ ; ⟨ список ссылок ⟩

⟨ числовые данные ⟩ :: = ⟨ список значений ⟩ | ⟨ числовые данные ⟩ ; ⟨ список значений ⟩

⟨ список значений ⟩ :: = ⟨ арифметическое выражение ⟩ | ⟨ список значений ⟩ , ⟨ арифметическое выражение ⟩

⟨список целых⟩:: =⟨целое⟩|⟨список целых⟩,⟨целое⟩|⟨список целых⟩;⟨целое⟩

⟨список ссылок⟩ - см. с. 75.

Оператор присваивания служит для выполнения определенных вычислений и присваивания значений арифметического или геометрического выражения переменным.

⟨оператор присваивания⟩:: =⟨идентификатор⟩=⟨выражение⟩#

⟨выражение⟩:: =⟨арифметическое выражение⟩|⟨геометрическое выражение⟩

Например:

```
ALF = 2.5•COS(X)+LN(ABS(A+1))•42.3#

T = ТОЧКА (T2; B1; 3.2)#
```

Здесь переменной ALF присваивается значение арифметического выражения, стоящего в правой части; переменная *T* становится точкой, которая отстоит от точки *T2* в направлении вектора *B1* на расстоянии 3,2. Условимся в дальнейшем в конструкциях геометрических выражений писать маленькими буквами тип геометрического объекта, подразумевая под этим идентификатор этого геометрического объекта. Тогда предыдущий оператор присваивания запишется так:

```
T = ТОЧКА (точка; вектор; 3,2)
```

Оператор присваивания, в правой части которого находится геометрическое выражение, в первую очередь вычисляет значения элементов списка параметров, затем вычисляет сам геометрический объект, и, наконец,ему присваивается идентификатор переменной, находящейся в левой части оператора. Значением геометрического объекта являются числовые данные о нем.

Применяя оператор эквивалентности,можно одному и тому же геометрическому объекту или простой переменной дать два наименования. Использование этого оператора позволяет экономить оперативную память машины. Следует особо отметить, что оператор эквивалентности применим только к о о ъ е к т а м о д н о г о и т о г о ж е т и п а.

⟨оператор эквивалентности⟩:: =⟨идентификатор⟩ @ ⟨идентификатор⟩#

5. Процедуры

Под процедурой в языке понимается определенная последовательность специальным образом оформленных операторов, которая многократно используется с различными данными. Язык позволяет обращаться к процедурам (т.е. получать результат выполнения процедуры) из различных мест программы, в которой эти процедуры описаны. Об -

ращению к процедуре должно предшествовать ее описание. Не допускается помещать описание одной процедуры в описание другой.

⟨ описание процедуры ⟩:: = ПРОЦЕД ⌴ ⟨ заголовок процедуры ⟩ # ⟨ тело процедуры ⟩ # КОНЕЦП #

⟨ тело процедуры ⟩:: = ⟨ оператор ⟩ | ⟨ тело процедуры ⟩ # ⟨ оператор ⟩

⟨ оператор⟩ :: = ⟨ оператор присваивания ⟩ | ⟨ оператор эквивалентности ⟩ | ⟨ оператор обращения к процедуре ⟩

⟨ заголовок процедуры ⟩:: = ⟨ идентификатор процедуры ⟩ (⟨ список формальных параметров ⟩) | ⟨ идентификатор процедуры ⟩ (⟨ список формальных параметров ⟩) # ⟨ спецификация ⟩

Заголовок процедуры делится на две части, из которых правая часть (идентификатор процедуры со списком формальных параметров) является обязательной, а вторая часть (спецификация) может отсутствовать.

⟨ список формальных параметров ⟩:: = ⟨ идентификатор ⟩ | ⟨ идентификатор⟩ = ⟨ основное служебное слово ⟩ | ⟨ список формальных параметров ⟩, ⟨ идентификатор⟩ = ⟨ основное служебное слово ⟩ | ⟨ список формальных параметров ⟩, ⟨ идентификатор ⟩

Каждый идентификатор списка формальных параметров специфицируется, т.е. указывается его тип.

При выполнении тела процедуры некоторые фактические параметры должны участвовать непосредственно в виде тех выражений, которыми они заданы при обращении к процедуре. В других случаях, наоборот, для выполнения процедуры достаточно знать только значения фактических параметров, изменение которых при выполнении процедуры нежелательно. Синтаксис описания процедуры позволяет выделить в отдельный список те формальные параметры, для которых при обращении к процедуре важно знать только значения соответствующих фактических параметров. Под параметры, описанные в спецификации, при трансляции процедуры отводится специальная рабочая область, в которую заносятся скопированные значения этих параметров.

⟨ спецификация ⟩:: = VALUE : ⟨ список параметров value ⟩

⟨ список параметров value ⟩ :: = ⟨ идентификатор ⟩ | ⟨ список параметров value ⟩ , ⟨ идентификатор ⟩

Переменные, встречающиеся в операторе тела процедуры, могут быть либо формальными параметрами, либо локальными переменными.

Локальная переменная используется только внутри процедуры, и ее значение недоступно для вызывающей прог-

раммы. Совпадение имени локальной переменной с именем какой-либо переменной вызывающей программы не является ошибкой.

Под п а р а м е т р о м понимаются идентификаторы, которые встречаются в списке формальных параметров при описании заголовка процедуры.

Необходимо, чтобы в обращении к процедуре число фактических параметров совпадало с числом формальных параметров в описании заголовка процедуры и при этом соблюдалось их позиционное соответствие. Синтаксис обращения к процедуре имеет следующий вид:

⟨обращение к процедуре⟩:: =⟨идентификатор процедуры⟩ (⟨список фактических параметров⟩)#

⟨список фактических параметров⟩:: = ⟨фактический параметр⟩ | ⟨список фактических параметров⟩,⟨фактический параметр⟩

⟨фактический параметр⟩:: =⟨идентификатор⟩|⟨арифметическое выражение⟩|⟨целое⟩

Фактические параметры типа VALUE обязательно должны быть описаны до обращения к процедуре. Кроме того, должны быть описаны и те параметры, которые первый раз встречаются в правой части оператора в теле процедуры. Для остальных параметров описание не обязательно.

Любая процедура, описанная в тексте транслируемой программы, может быть каталогизирована в библиотеку. Для этого непосредственно перед описанием процедуры вставляется оператор

ЗАПИСАТЬ ⟨ИМЯ ПРОЦЕДУРЫ⟩ #

Для включения библиотечной процедуры в текст программы надо в соответствующем месте программы до первого обращения к ней вставить оператор

ВКЛЮЧИТЬ ⟨ИМЯ ПРОЦЕДУРЫ⟩ #

Для удаления процедуры из библиотеки используется оператор

УДАЛИТЬ ⟨ИМЯ ПРОЦЕДУРЫ⟩ #

6. Перспективы развития системы

Дальнейшее развитие системы на первый план выдвигает задачи создания средств мультидоступа, совершенствования системы управления архивами, входного языка и создания средств расширения языка. Структура математического обеспечения системы изображена на рис.2.

В состав программных средств входят: программа инициализации системы, осуществляющая подготовку различных таблиц системы и областей памяти, настройку и запуск диспетчера; диспетчер системы, осуществляющий прием и предварительную обработку запросов с терми-

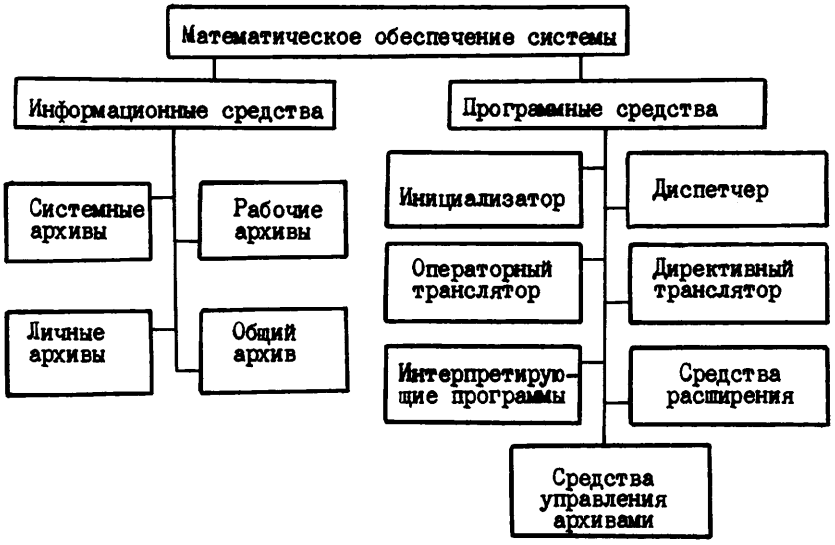

Рис.2

нала и подчиненных программ; транслятор с операторной части языка, осуществляющий семантический анализ входной программы, формирование обращения к соответствующим интерпретирующим программам, подготовку абсолютной программы, реализующей оператор; транслятор с директивной части языка, осуществляющий семантический анализ поступающих директив, их исполнение и отсылку к диспетчеру; набор интерпретирующих программ, реализующих конкретные функции операторов и директив; средства расширения транслятора, которые преобразуют соответствующие таблицы и дополняют набор семантических подпрограмм трансляторов; средства управления архивами, которые обеспечивают доступ к хранимой информации.

Информационное обеспечение системы представляет собой совокупность системных, рабочих и личных архивов, архив общего пользования. Системный архив создается в момент вхождения пользователя на время его работы с системой. Он предназначен для хранения вводного текста пользователя, хранения диагностических и других сообщений трансляторов. Рабочий архив создается системой в момент трансляции программы пользователя и предназначен для хранения про-

межуточной информации и результатов работы интерпретирующих модулей. Системный и рабочий архивы создаются для каждого пользова - теля без его ведома и являются внутренними информационными хранилищами системы.Личные архивы создаются по желанию пользователя и предназначены для долговременного хранения информации. Архив общего пользования создается в момент генерации системы и предназначен для хранения информации,доступной всем пользователям, имеющим шифр доступа. Заносить информацию в этот архив разрешено определенному лицу.

Входной язык системы условно разбивается на две части: директивную и операторную. Директивная часть языка дает пользователю возможность производить различные преобразования текста программы (и произвольного текста), получать информацию о состоянии своего процесса.

Работа с произвольным символьным текстом задается директивой ПОСТ, которая все последующие директивы интерпретирует как команды обработки символьного текста до появления директивы ПРОГ. Эта директива настраивает последующие на обработку текста, имеющего структуру операторной части входного языка. Таким образом, директивы ПОСТ и ПРОГ задают режимы обработки входного текста. В каждом режиме используются директивы: УДАЛИТЬ, ВСТАВИТЬ, ЗАМЕНИТЬ, ПЕЧАТЬ, ПЕРФОРАЦИЯ, ТИРАЖ, &*, СОХРАНИТЬ,ЧИТАТЬ, КОНЕЦ, ВЫДАТЬ, СОСТОЯНИЕ, ОТКРЫТЬ ТЕРМИНАЛ, ЗАКРЫТЬ ТЕРМИНАЛ. Кроме того,в режиме ПРОГ используются директивы ПЕРЕНУМЕРОВАТЬ и ПУСК, а в режиме ПОСТ - директивы НУМЕРОВАТЬ и &.

Операторная часть предназначена для написания программ.Любая программная единица состоит из набора операторов. Операторы компонуются из ключевых слов, используемых совместно с основными элементами языка. К основным элементам языка относятся константы (числовые, логические, литеральные), переменные,функции, выражения (арифметические, логические, типовые) и другие. Ключевые слова определяют вид и назначение оператора.Некоторые операторы компонуются без использования ключевых слов.

Операторы разделяются на выполняемые и невыполняемые. Выполняемые определяют действия. К ним относятся операторы присваивания, операторы управления, операторы ввода/вывода и манипулирования данными. К невыполняемым относятся операторы объявления начала и конца программы, начала и конца процедуры, спецификации параметров процедуры, эквивалентности, а также технологические параметры, условия обработки и другие.

Программные средства реализуются в настоящее время в рамках операционной системы ОС ЕС ЭВМ.

Литература

I. ЧЕРЕПАНОВ Е.М., КОНСТАНТИНОВ В.И. Программное обеспечение системы автоматизации проектирования и подготовки технологических данных. - Отчет ИМ СО АН СССР, 1975.

2. Автоматизация проектирования и управления технологическими процессами в машиностроении / Гамидов В.М., Миренкова Т.П., Овчинникова Т.Э., Харченко В.П. - Отчет ИМ СО АН СССР, 1980.

3. ЗАВЬЯЛОВ Ю.С., КВАСОВ Б.И., МИРОШНИЧЕНКО В.Л. Методы сплайн-функций. - М., Наука, 1980. - 352 с.

Поступила в ред.-изд.отд.
24 апреля 1981 года