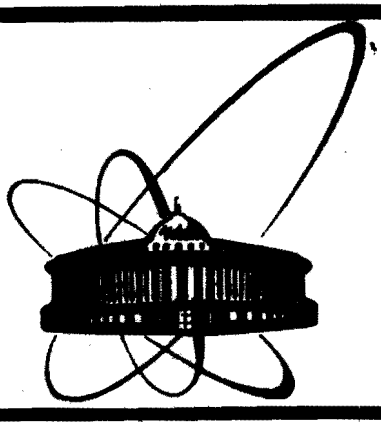

сообщения
объединенного
института
ядерных
исследований
дубна

Р11-87-939

В.Тодоров

АВТОМАТИЗИРОВАННАЯ СРЕДА ДЛЯ РАЗРАБОТКИ ДИАЛОГОВЫХ ПРИКЛАДНЫХ СИСТЕМ

1987

При разработке автоматизированных систем часто можно наблюдать несоответствие возможностей системы истинным потребностям пользователей, особенно при взаимодействии человека с ЭВМ.

Эта ситуация возникает, как правило, на этапе внедрения системы. Причины ее, в частности, таковы:

I. Пользователь не совсем ясно представлял себе, что ему нужно.

2. Разработчик, в силу своей компетентности главным образом в области программ и баз данных, уделяет им внимание и оставляет в стороне проблемы взаимодействия с ЭВМ.

В системах пакетной обработки острота этой проблемы смягчается тем, что в эксплуатацию вовлекаются звенья, выполняющие подготовку и ввод данных в ЭВМ, интерпретацию полученных результатов для пользователя, а также некоторую их обработку, принимая тем самым на себя некоторые функции, не реализованные в системе.

При широком распространении диалоговых систем пользователь непосредственно сталкивается с ЭВМ, минуя промежуточные звенья, и вынужден сам вводить данные и разбираться в результатах. Становится невозможным скрыть несоответствие системы его требованиям. Это - одна из многих особенностей диалоговых систем.

Другой особенностью диалоговых систем является соотношение объемов программного обеспечения, обслуживающего диалог и обеспечения, реализующего прикладные функции системы. Как правило, диалоговая часть в несколько раз превосходит по объему прикладную часть.

Эти особенности диалоговых систем выступают предпосылками метода опережающего проектирования и моделирования диалога при разработке прикладных диалоговых систем /3/. Проект "Автоматизированная среда для разработки диалоговых прикладных систем" (АСРПДС) выступает как инструмент поддержки этого метода. В основе проекта - выделение пакета подпрограмм диалоговой части системы и отделение от них описания диалога (сценария). Программы интерпретируют сценарий, организуя ввод/вывод данных с/на терминал и обращаясь к модулям прикладной части для функциональной обработки. Прикладные модули программируются отдельно в процессе разработки прикладной системы и подключаются к пакету с помощью средств операционной системы. Реализуя при опережающем про-

ектировании и моделировании диалога функции прикладных модулей в разном объеме, можно получить варианты прикладной системы в диапазоне от действующей модели системы до системы в полном объеме запроектированных функций. Это позволит увидеть на действующей модели (макете) будущей системы недостатки проекта, а после согласования проекта диалога - использовать эту модель как тренажер для пользовательского персонала. Сократится также и период разработки прикладной части системы, для чего в пакете предусматриваются средства создания/корректировки сценария диалога и воспроизведения диалога по сценарию, а также средства подготовки документации на прикладную часть системы.

Ниже рассмотрена реализация проекта АСРПДС (I-я версия). Вводятся понятия видеограммы, сегмента, обмена, последовательностей обменов и реакции системы. Излагается механизм работы на базе идеи основного цикла системы. Дается краткая характеристика архитектуры математического обеспечения АСРПДС.

Система предназначена для проектирования диалоговой части мат. обеспечения прикладных диалоговых систем. Она должна решать следующие задачи:

I) создание динамической модели диалога и внесение изменений в эту модель;

2) создание документации для разработки мат.обеспечения прикладной части системы;

3) подключение к модели подпрограмм, реализующих функции разрабатываемой системы (функциональных модулей).

Система задумана как инструмент для персонала, разрабатывающего прикладные системы. Она ориентирована на взаимодействие человека с ЭВМ, осуществляемое через видеодисплей и клавиатуру. В ответ на ввод с клавиатуры система редактирует данные, подготовленные функциональными модулями, и выводит их на экран. Формат редактирования определяется разработчиком еще на этапе создания модели диалога. Тогда же определяется и логика взаимодействия человека с ЭВМ - какой выход подготовить и как его отредактировать в ответ на определенный вход. Запись форматов и логики взаимодействия образуют сценарий диалога, который хранится во внутреннем представлении в базе данных системы.

ОСНОВНЫЕ ПОНЯТИЯ И ИДЕИ СИСТЕМЫ

Видеограммы и сегменты

С точки зрения пользователя диалог состоит в построении картин на экране. Выдав на экран порцию информации, система ждет ответа пользователя, который тот вводит с алфавитно-цифровой клавиатуры. На ввод система отвечает дополнением картины, существующей на экране, или построением новой. Выводимые данные выбираются по определенным правилам. Эту элементарную часть диалога назовем обменом /I/.

Картины на экране между отдельными его зачистками назовем видеограммами. Элементы картин, выводимые в ответ на ввод пользователя, назовем сегментами видеограмм. Точнее, сегментом назовем прообраз, по которому строится картина на экране. При построении он дополняется текстом выходов (прикладных данных, выработанных системой в качестве ответа пользователю). Сегменты выбираются по определенным правилам посредством т.н. реакции системы. Для этого в сценарий диалога входят также и указания, в какой последовательности выбирать сегменты для последовательных обменов. Управление диалогом основано на механизме, предполагающем предварительное проектирование возможных вычислительных ситуаций, связывание с каждой из них подходящего продолжения и возлагающем на прикладную часть распознавания ситуации и выбор соответствующего продолжения.

Последовательность обменов

Решая задачу, пользователь может столкнуться с ситуацией, когда ему неизвестны какие-то данные и для продолжения работы по основной задаче он должен их уточнить.

Нужная информация находится в системе, но указать ее может только пользователь, возможно, с помощью системы при формулировке ввода. В конечном счете решение должен принять пользователь. Для этого позволяется прервать процесс решения основной задачи, не отказываясь от нее, и начать решать некую вспомогательную задачу. С помощью системы пользователь узнает информацию, недостающую для решения основной задачи, и может продолжить работу.

При переходе к дополнительному диалогу система не теряет связи с основным диалогом. Основной диалог прерывается, но возможность вернуться к нему сохраняется. Для этого механизм управления включает элементы прерывания и возврата. Последовательность обменов можно разбить на две группы:

- основные (главные) последовательности;
- вспомогательные (дополнительные) последовательности.

Основные последовательности соотносятся с решаемыми в системе задачами. Они состоят из тех наименьших множеств шагов, через которые так или иначе надо пройти, решая данную задачу согласно сценарию диалога.

Последовательности обменов, связанных с решением вспомогательных задач, образуют вспомогательные (дополнительные) последовательности. Схематично это изображено на рис.I, где вспомогательная последовательность разрывает основную.

Реализуются два вида перехода от обмена к обмену:

- обычный, когда в диалоге проходим основную последовательность;
- прерывание, когда переходим к обмену не из основной последовательности.

Возможны ситуации: а) система не может помочь пользователю, хотя в базе данных существует необходимая информация, но принять решение, сделать выбор, сформулировать нужный ответ может только пользователь; б) система прямо участвует в формулировке ответа. В результате решения дополнительной задачи система получает (вычисляет) нужный текст, и пользователю нет необходимости его вводить.

Для этого вводим две формы завершения вспомогательной задачи: 1) возврат в точку прерывания для ввода необходимого ответа; 2) переход к обмену, следующему за прерванным в основной задаче, с использованием ответа, подготовленного системой.

Первую форму назовем "возвратом с повторением ввода", вторую - "возвратом с продолжением диалога". Им соответствуют диаграммы на рис.2б и 2в.

Допускаем также прерывание и вспомогательных задач с переходом к другой вспомогательной задаче (т.е. прерывание на двух уровнях). Пути возврата в основную последовательность будут:

1-й непосредственно в основную последовательность;

2-й на предыдущий уровень и оттуда уже в основную последовательность.

При возврате возможны обе формы - как с повторением ввода, так и с продолжением диалога. В данной реализации допускаются прерывания на многих уровнях.

Наконец, отметим, что протекание диалога можно рассматривать как проход по узлам графа, представляющего сценарий диалога. При реализации алгоритмов обхода графа полезным оказывается применение стековых механизмов.

Программы диалоговой части передают прикладные данные от пользователя программам прикладной части (т.н. функциональным модулям или сокращенно - ф.модулям) и выводят на экране ответы системы, редактируя их в соответствии с описанием сегмента. В зависимости от сложившейся ситуации эти программы выбирают описание сегмента в соответствии с реакцией системы.

Осуществляя обмены, диалоговая компонента циклически повторяет следующую последовательность действий:

1) получение от ф.модуля данных и реакций,
2) выбор сегмента,
3) редактирование данных сегмента,
4) вывод сегмента на экран,
5) получение данных с клавиатуры,
6) передача данных ф.модулю.

Эту последовательность действий мы назовем "основным циклом". В течение одного цикла выполняется один обмен между пользователем и системой. В картине на экране некоторые поля изменяются (системой или пользователем). Будем говорить о переменных полях в сегменте и, соответственно, об описании переменных полей в описании сегмента.

ПРИНЦИПЫ РЕАЛИЗАЦИИ

Основной цикл диалога и представление реакций

В основном цикле диалога создаются и данные, определяющие следующий обмен ("заготовку" текста для вывода и ф.модуль). Мы назвали такие данные "реакцией системы". Протекание диалога рассматривалось как последовательность циклов обмена, продолжающуюся до тех пор, пока в очередном обмене не будут введены данные, требующие завершения диалога.

Для реализации такого рода диалога предлагается следующий механизм. Его основным элементом является совокупность описаний сегментов. В описание отдельного сегмента входят:

- тексты отдельных строк сегмента;
- признак типа сегмента;
- ссылки на описания переменных полей (входов/выходов) сегмента;
- список реакций сегмента.

Описания полей задают правила превращения строки, выданной ф.модулем в текст, который будет выведен на экран дисплея, и указывают места на экране, куда вывести (либо откуда читать) прикладные данные, а также структуру этих данных.

Второй элемент механизма - описания реакций. Это - список команд, определяющих действия диалоговой части системы по выбору следующего обмена: сегмента, который определит текст на экране, и ф.модуля, который обслужит обмен, а также характер манипуляций с текстом текущего сегмента.

Ф.модуль, обслуживающий текущий сегмент, свою реакцию выражает, указывая номер элемента в этом списке.

Команды реакций состоят из двух числовых частей и кода. Первое число определяет ф.модуль, который обслужит следующий обмен, интерпретация второго числа зависит от кода команды.

Команды определяют обычные переходы, прерывания и возвраты. Структура команд и их действие даны на рис.3. В описаниях команд используется обозначение команды в виде упорядоченной тройки (M, S, C), где

M - номер модуля,который обслужит обмен;
S - номер сегмента, который определит контекст обмена;
C - символ, определяющий тип реакции.

Команда выполняется после очередного обмена (его называем предыдущим обменом), когда получены прикладные данные от клавиатуры (ввод, входная строка), и перед тем, как обратиться к прикладной программе для их обработки.

Третьим компонентом рассматриваемого механизма является стек, где запоминается состояние диалога в момент прерывания и откуда оно извлекается для восстановления при возврате. Это делается классическими операциями "PUSH" и "POP" и задается командой реакции. Состояние диалога определяется текстом сегмента, модифицированным прикладными данными в ходе обмена, а также значением системных данных. Возврат на N уровней осуществляется N-кратным сбрасыванием стека.

Таким образом, для задания обмена система располагает источниками, которые полностью определяются кодом команды реакции: 1) множество "заготовок", 2) текущее состояние, 3) стек, 4) рабочая область в программе.

Способ редактирования задается типом сегмента, тем самым определяя обмен полностью.

Виды сегментов

Мы различаем 3 типа сегментов: 1) сегмент фиксированный, 2) сегмент динамический ("окно"), 3) сегмент динамический ("страница"). Следует предусмотреть возможность вводить новые типы сегментов.

Тип 1: сегмент фиксированный.

Текст, выводимый на экран, и его формат определяются не форматом и текстом, выведенным на экран в предыдущем обмене, а только реакцией системы на ввод в предыдущем обмене. Для вывода в текущем обмене используется новая "заготовка".

Тип 2: сегмент динамический ("окно").

Текст, выводимый на экран, и его формат определяются сегментом предыдущего обмена. Для вывода в текущем обмене используется и текст, сформированный в предыдущем обмене и модифицированный выходными данными текущего обмена.

Тип 3: сегмент динамический ("страница").

Только формат текста, выводимого на экран, определяется сегментом предыдущего обмена. Для вывода в текущем обмене используется текст, полностью сформированный функциональным модулем предшествующего обмена.

Дадим примеры сегментов второго и третьего типов.

В сегменте типа "окно" реализуется построчный вывод со сдвигом вверх, как в терминалах телетайпного типа. Описан формат строки, общей для всех строк в сегменте.

В сегментах типа "страница" выходы тоже описывают одну единственную строку, но редактируются сразу все строки сегмента. При повторном обращении в следующем обмене к этому же сегменту все его строки будут вновь отредактированы ("перелистывание").

Входы в сегментах такого типа не предусматриваются, и реакция ф.модуля должна указывать на сегмент, который обеспечит вход.

Виды переменных полей

В тексте сегмента на экране поля делятся на фиксированные и переменные. Фиксированные поля – это те, которые задаются при проектировании диалога и не меняются во время диалога. Изменять их может только дизайнер диалога, корректируя сценарий. Переменные поля – это поля, в которые заносятся прикладные данные либо пользователем с клавиатуры, либо системой. Эти поля описываются при проектировании диалога указанием места на экране и синтаксисом располагаемых здесь прикладных данных. Для задания синтаксиса прикладных данных в системе используются форматные спецификации. Простые спецификации описывают элементарные данные, составные спецификации вводятся для описания переменных полей, в которые заносятся несколько значений одновременно. Простые форматные спецификации определяют следующие типы данных: арифметические (целые и с дробной частью), символьные, литералы и даты. Форматные спецификации используются в данной реализации для определения длин полей при вводе и при выводе, а также для конверсии перед выводом на экран данных, полученных от ф.модулей. В дальнейшем они будут использоваться для проверки синтаксической правильности введенных данных.

Все данные, описывающие элементы этого механизма (описания сегментов, полей и т.д.), организованы в базу данных сценария (Б.Д.С.). Ее состав представлен на рис.4.

Для создания диалога для прикладной системы дизайнер должен представить себе процесс решения прикладной задачи в виде последовательности картин на экране, разделенной моментами ввода данных, и предусмотреть вычислительные ситуации с соответствующим продолжением диалога (реакции системы). Для этого дизайнер вводит тексты сегментов, описания переменных полей и списки троек (M,S,C).

В данной версии не реализованы концепции применения критериев удобства, намеченные в [3].

Структура математического обеспечения АСРПДС

Эта структура складывается из файлов Б.Д.С. и программ , работающих с этими файлами.

Файлы, в которых расположена Б.Д.С., включают:

SGM – тексты постоянных полей, ссылки на переменные поля и описания реакций;

IOD – описания переменных полей;

FMT – форматные спецификации.

Совокупность программ включает три комплекса:

- генератор сценария,
- интерпретатор сценария,
- генератор документации.

В данной экспериментальной версии реализация пока ограничена объемом до 1000 сегментов и до 100 ф.модулей. Использован язык программирования ПЛ/1. В качестве интерфейса с терминалами на нижнем уровне подключена подпрограмма TERMCP (элемент системы ТЕРМ). Для связи программ высшего уровня с TERMCP используется интерфейсная подпрограмма на Ассемблере. Система работает под управлением операционной системы ОС ЕС. Для переноса в другую терминальную среду нужно переписать интерфейсную подпрограмму, учитывая соответствующий терминальный драйвер.

Генератор сценариев предназначен создавать, удалять , модифицировать сценарии. Пользователь, работающий с ним, имеет специальные функции; в частности, он должен знать специфику работы пользователя будущей системы лучше, чем подробности программирования. В дальнейшем такого пользователя будем называть дизайнером диалога. В его задачи входит не оптимизация базы прикладных данных и прикладных программ, а оптимизация диалога.

Генератор сценариев работает в режиме диалога. Контролируются полнота и непротиворечивость данных сценария, так чтобы сценарий диалога в любой момент оставался согласованным. Дизайнер может в любой момент запросить имитацию любого фрагмента сценария, вплоть до всего диалога.

Для имитации диалога генератор сценариев обращается к интерпретатору, являющемуся вторым программным комплексом АСРПДС. Этот комплекс выполняет основной цикл и управляет диалогом, следуя командам реакций и обращаясь к фиктивным модулям. Имитация диалога позволяет легко осуществить итеративный процесс разработки прикладной системы, согласованный с принципами TOP-DOWN (т.е. от потребностей пользователя, а не от возможностей языков программирования и прочих средств разработки [4]).

Третий программный комплекс – это генератор документации. Его назначение – печатать проектную документацию будущей прикладной системы по запросу пользователя.

Документация имеет форму спецификаций ф.модулей, сопровождаемых перечнями прикладных данных, форматов, диаграмм сценария, макетов видеограмм. Предусматривается режим просмотра этих данных на экране и возможность получить актуальные экземпляры документации в виде бумажных копий.

Заключение

Рассматриваемая система задумана как инструмент проектировщика систем с диалоговым взаимодействием пользователя с ЭВМ. Она призвана решить хотя бы отчасти проблемы проектирования систем с помощью опережающего проектирования диалога [3], используя концепцию двухкомпонентной структуры диалоговых систем [2].

Такой подход дает определенные преимущества при проектировании систем. Работающий макет прикладной системы позволит пользователю быть в курсе продвижения разработки, обоснованно высказывать рекомендации к будущей системе, что весьма важно при разработке диалоговых систем [3]. Избегается конфликт, связанный с пренебрежением мнением пользователя [2]. При традиционных методах разработки пользователь, не будучи специалистом в области программирования и разработки диалоговых систем, не в состоянии вникнуть в возникающие проблемы разработки. АСРПДС исправляет это положение, предоставляя на достаточно ранних этапах пользователю завершенный диалог, не требуя для этого разработки функциональной части прикладной системы и понимания пользователем ее потребностей.

Разработчик программ освобождается от необходимости думать о подробностях реализации диалога, но ему оставлена полная свобода в выборе структуры прикладной базы данных, в вопросах оптимизации алгоритмов доступа и обработки и т.д.

Вкратце преимущества подхода можно резюмировать так:

1) определенность для пользователя: создав совокупность описаний сегментов (сценарий) и пользуясь заглушками вместо реальных ф.модулей, можно получить действующий макет/модель;

2) определенность для разработчика: представив надлежащим образом сценарий, можно получить спецификацию модулей прикладной компоненты системы (функциональных модулей);

3) ускорение разработки: при разработке ф.модулей уже не нужно заботиться о взаимоотношениях с экраном; их объем уменьшится сильно, и запрограммировать их будет легче.

Основные последовательности.

Это – последовательности обменов, в ходе которых пользователь решает ту задачу, ради которой и была создана прикладная система.

Дополнительные (вспомогательные) последовательности.

При невозможности сформулировать свой ввод пользователь может вступить в другой диалог с системой для нахождения каких-то данных или принятия решения (дополнительная задача), решение дополнительной задачи здесь либо вырабатывает искомый ввод, либо позволяет пользователю выработать его.

$\ldots \rightarrow E_o \rightarrow E_1 \rightarrow \ldots \rightarrow E_I \rightarrow E_J \rightarrow E_{J+1} \rightarrow \ldots \rightarrow E_{I+1} \rightarrow$

Обмены:

E_o , E_1 ,... E_I, E_{I+1},... – образуют главную последовательность

E_J , E_{J+1},... E_{J+K} ... – образуют вспомогательную последовательность

Рис.I. Основные и вспомогательные последовательности обменов.

а) Основная последовательность диалога

$\ldots \rightarrow E_I \rightarrow E_{I+1} \rightarrow \ldots$

б) Прерывание и возврат с повторением ввода

$\ldots \rightarrow E_I \rightarrow E_J \rightarrow E_{J+1} \rightarrow \ldots \rightarrow E_{J+K} \rightarrow E_I \rightarrow E_{I+1} \rightarrow \ldots$
прерыв. возвр.

в) Прерывание и возврат с продолжением диалога

$\ldots \rightarrow E_I \rightarrow E_J \rightarrow E_{J+1} \rightarrow \ldots \rightarrow E_{J+K} \rightarrow E_{I+1} \rightarrow \ldots$
прерыв. возвр.

Рис.2. Основные и вспомогательные последовательности обменов. Виды возврата в основную последовательность.

Обычные переходы

(M, S,'N') Выбрать "заготовку" сегмента S и начать новую видеограмму (зачистить экран); полученные от клавиатуры прикладные данные передать модулю M .

(M, S,' ') Выбрать "заготовку" сегмента S ; полученные от клавиатуры прикладные данные передать модулю M .

(M,,'R') Отредактировать текущий сегмент (т.е. не выбирать новой "заготовки"); полученные от клавиатуры прикладные данные передать модулю M .

(M,,'D') Вывести диагностическое сообщение в месте, определенном в описании текущего сегмента (т.е. не выбирать новой "заготовки"); полученные от клавиатуры прикладные данные передать модулю M .

(M,,'E') Передать управление модулю M (как правило, для завершения диалога).

Прерывания
(текущее состояние диалога запоминается в стеке)

(M,S,'W') После запоминания в стеке выбрать "заготовку" сегмента S и начать новую видеограмму (зачистить экран), полученные от клавитауры прикладные данные передать модулю M .

(M,S,'S') После запоминания в стеке выбрать "заготовку" сегмента S ; полученные от клавиатуры прикладные данные передать модулю M .

Возврат в основную последовательность

(M,N,'T') Возврат с повторением ввода. Если N=0 , то на предыдущий уровень, если N<>0 , то на (N+1) уровней назад. (M,,'T') эквивалентно (M,0,'T').

(M,N,'C') Возврат с продолжением диалога. Если N=0 , то на предыдущий уровень, если N<>0 , то на (N+1) уровней назад. (M,,'C') эквивалентно (M,0, 'C').

Рис.3. Команды реакций.

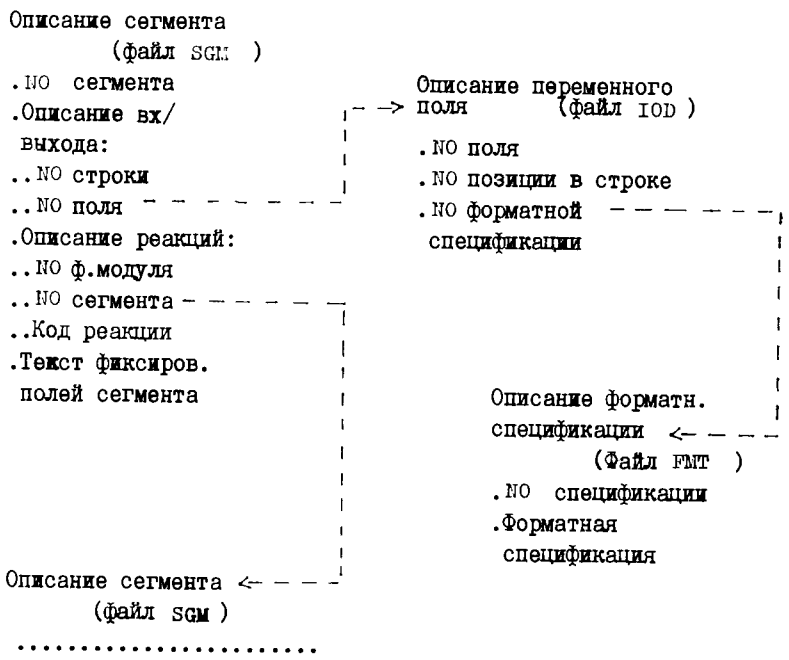

Рис.4. Структура базы данных сценария.

Литература

I. Блэкман М. Проектирование систем реального времени. М., Мир,I980.

2. Денинг В., Эссиг Г., Маас С. Диалоговые системы "Человек-ЭВМ". Адаптация к требованиям пользователей. М., Мир, I985.

3. Тодоров В. ОИЯИ, PII-87-938, Дубна, I987.

4. Gaines B.R., Shaw M.L.G. From timesharing to the sixth generation: the development of human-computer interaction. part I. International journal of man-machine studies, 1986, 24, p. 1-27.

Рукопись поступила в издательский отдел
30 декабря I987 года.

Тодоров В. P11-87-939

Автоматизированная среда для
разработки диалоговых прикладных систем

Рассматривается представление диалога в виде последовательных обменов через алфавитно-цифровые клавиатуру и дисплей. Вводятся понятия видеограмм, реакций, основного цикла обмена и т.п. На их базе описывается строение и функционирование средств для обслуживания взаимодействия прикладной системы с пользователем. Добавление средств создания, сопровождения, документирования диалога позволяет говорить об автоматизированной среде для разработки прикладных диалоговых систем.

Работа выполнена в Лаборатории вычислительной техники и автоматизации ОИЯИ.

Сообщение Объединенного института ядерных исследований. Дубна 1987

Перевод О.С.Виноградовой

Todorov V. P11-87-939

Automatized Environment Applied Dialogue
System Development

A dialogue representation as a sequence of exchanges by means of an alphabetical keyboard and CRT is considered. The concepts of panels, reactions, foreground exchange cycle and so on are introduced. Based on these concepts the service for means an user applied system interaction structure and functioning are described. Adding the means for creating/supporting/documenting a dialogue give a possibility to concern the complex as an automatized environment for the applied dialogue system development.

The investigation has been performed at the Laboratory of Computing Techniques and Automation, JINR.

Communication of the Joint Institute for Nuclear Research. Dubna 1987