Лекция № 5.
Процессы ввода-вывода как средство осуществления диалога.

Термин «процесс» используется для описания последовательности операций, выполняемых системой.

Термин «задание» - для обозначения того, что хочет сделать пользователь. Например, задание по вычислению средней прибыли группы товаров. Теоретически существует бесконечное множество заданий, которые нужно выполнить; на практике большинство систем имеет дело лишь с ограниченным числом процессов

Каждое задание может быть выполнено одним или несколькими процессами. Не существует однозначного соответствия между заданиями и процессами. Один процесс может использоваться для выполнения нескольких заданий; одно задание может выполняться несколькими процессами.

При разработке интерфейсов процессы рассматриваются как «черные ящики». В общем случае их называют процессами по выполнению задания или просто заданиями.

Интерфейс человек-компьютер обеспечивает связь между пользователем и процессом, выполняющим задание. Это дает возможность пользователю определять, какие задания сделать активными в данный момент, как передавать им данные для обработки и принимать результаты обработки. С точки зрения программного обеспечения в состав интерфейса входят два компонента: процесс диалога и набор процессов ввода-вывода.

Пользователь компьютерной системы взаимодействует с интерфейсом: через интерфейс он посылает входные данные и принимает выходные. Процессы по выполнению заданий вызываются интерфейсом в требуемые моменты времени.

Процессы ввода-вывода обеспечивают обмен информацией на самом верхнем уровне процесса диалога. На этом уровне диалоговый процесс должен правильно интерпретировать каждое слово и даже каждый звук.

***Задачи диалогового процесса*:**

- определение задания, которое пользователь возлагает на систему;
- прием логически связанных входных данных от пользователя и размещение их в переменных соответствующего процесса в нужном формате;
- вызов процесса выполнения требуемого задания;
- вывод результатов обработки по окончании процесса в подходящем для пользователя формате.

Процессы ввода-вывода.

Процессы ввода-вывода служат для того, чтобы принять от пользователя и передать ему данные через различные физические устройства. При выборе устройств учитываются следующие факторы.

Содержание и формат обрабатываемых данных. Для некоторых прикладных задач нужен лишь ограниченный диапазон текстовых символов, для других – графический режим с высокой разрешающей способностью. Иногда пользователь должен вводить набор произвольных величин, а иногда нужно сделать только выборку из небольшого набора возможных значений.

Объем ввода-вывода. Увеличение объема входных данных предполагает наличие косвенного механизма ввода, например автоматического сбора данных.

Ограничения, накладываемые пользователем и рабочей средой. (Физические дефекты пользователя, работа в цеху, совмещение работы за компьютером с другими видами работ).

Ограничения, связанные с другими аппаратными и программными средствами, которые используются в системе.

С каждым устройством связан свой процесс ввода-вывода, задача которого – воспринять данные от пользователя и преобразовать их во внутреннее представление, с которым может работать процесс диалога.

Очевидно, что процесс ввода с клавиатуры и процесс речевого ввода различаются типами обрабатываемых ими устройств. Отделяя физический процесс ввода-вывода от процесса диалога, можно добиться того, что смена устройств ввода-вывода не приведет к изменению процесса диалога, а сведется лишь к замене процесса ввода-вывода.

Сообщения.

В диалоге информация передается в виде сообщений. В любом диалоге существует несколько типов сообщений.

Входные сообщения: команда, данные.

Выходные сообщения: подсказка, данные, состояние, ошибка, справка.

***Входные сообщения*.**

Команда – это входное управляющее сообщение, вызывающее процесс выполнения задания или какую-нибудь функцию диалогового процесса.

Входные данные – это данные, передаваемые пользователем процессу выполнения задания.

Могут быть и сложные сообщения, которые за один сеанс ввода вызывают нужный процесс и вводят данные.

Обычно для диалогового процесса нужно проверять введенные пользователем данные на наличие ошибок. Характер проверки зависит от формата входного сообщения. Управляющее сообщение проверяется на совпадение с одним из элементов списка возможных значений. Входные данные проверяются на нахождение их в пределах допустимого диапазона.

Введенное пользователем сообщение преобразуется диалоговым процессом в стандартный формат и передается на вход другого процесса. Это задание возвращает диалоговому процессу либо подтверждение о получении входных данных, либо результат обработки. Диалоговый процесс, в свою очередь, преобразует это выходное сообщение в подходящий для пользователя формат и выводит его в виде данных или как сообщение о состоянии системы.

Диалог можно классифицировать с учетом формата входных сообщений и гибкости, позволяющей пользователю вводить входные сообщения, когда ему удобно. Входное сообщение позволяет:

- выбрать режимы диалога, например, получение справки;
- выбрать нужный процесс выполнения задания;
- ввести данные для выполнения задания.

Формат, в соответствии с которым пользователь вводит свои сообщения, называют *грамматикой диалога* (коды, цепочки ключевых слов, ограниченный ЕЯ, ЕЯ).

Независимо от грамматики или способа ведения диалога, в его основе лежит следующий ***цикл***:

Явный или неявный запрос на ввод данных
Ввод данных через процесс ввода
Проверка входных данных

Этот цикл повторяется, пока не будут приняты приемлемые входные данные.

Если выводится запрос на ввод команды, следующий шаг обработки будет зависеть от введенной команды.

Рассмотрим теперь ***выходные сообщения*.**

Подсказка – это выходное сообщение системы, побуждающее пользователя вводить данные.

Реакцией пользователя на подсказку может быть входное управляющее сообщение или входные данные.

Существует ряд форматов вывода подсказок в диалоге.

Самый сложный формат – *меню*, когда наряду с запросом на ввод сообщения выводятся допустимые форматы ввода.

Выводить меню в виде текста необязательно. Можно вывести пиктограммы различных возможностей. Формат меню правомерен в том случае, если диапазон правильных входных данных не слишком велик и может быть явно выведен на экран. С помощью меню обычно выводятся команды системы.

Система может с помощью *вопроса* уточнить, какой тип данных требуется. Подсказка может содержать указание на требуемый формат входного сообщения.

Если требуется ввести данные сложного формата, например, даты, можно использовать в качестве подсказки специальную *форму* ввода.

Еще один вид подсказки – *запрос на ввод команды* (данных).

Выходные данные – это данные, которые возвращает диалоговый процесс по окончании обработки.

Сообщение о состоянии системы – это информация для пользователя о том, что произошло или происходит в системе.

Сообщение об ошибке – это сигнал диалогового процесса о том, что невозможно дальнейшее выполнение работы, т.к. вызванный процесс выполнения задания не может обработать сообщение, введенное пользователем.

Справочная информация – выводится в тех случаях, когда пользователь не может ответить на запрос системы, потому что ему непонятен запрос или он забыл, что нужно вводить. Справочная информация поясняет, что делать дальше и почему.

Ввод-вывод информации различных типов.

Классифицируем сообщения по характеру информации:

Вывод текстового сообщения:

- в текущую позицию на устройстве;
- в заданную позицию на устройстве;
- с заданием конкретного формата отображения.

Ввод текстового сообщения:

- с использованием стандартных процедур ввода;
- в режиме посимвольного ввода;
- с использованием управляющих символов.

Ввод сообщений типа Указать и Выбрать:

- просмотр предлагаемого списка операций и выбор нужной;
- выбор данных из любого места экрана.

Вывод графического сообщения.

Ввод графического сообщения.

Текстовое сообщение – это строка символов.

Графическое сообщение – это сообщение, которое нельзя представить в виде строки символов. Обычно описывается на битовом уровне.

Сообщения типа «Указать и Выбрать» – особый тип входных данных для обеспечения выбора из допустимого множества возможных альтернатив.

Управляющие сообщения обычно включают выбор из некоторого ограниченного множества вариантов. Входные данные содержат информацию о таком выборе.

Возможные альтернативы представляются списком выбираемых объектов. Такими объектами могут быть отдельные символы, строки текста, фрагменты экрана и т.д.

Каждый объект определяется следующими характеристиками.

Содержание – описывает объект; каждый объект должен однозначно определяться, чтобы пользователь знал, к чему обращаться.

Поле, в пределах которого объект отображается на экране; поля должны быть достаточно широкими и хорошо отделены друг от друга (всегда легче указать на большой объект, нежели на маленький).

Множество атрибутов, которые описывают его формат; область экрана, в которой находится объект, должна быть отчетливо выделена.

Число объектов не должно быть большим, чтобы пользователь не терялся при выборе. Существуют два основных способа указания пользователем определенного объекта: абсолютный или относительный выбор.

Абсолютный выбор позволяет пользователю указать любое место на экране независимо от того, размещается там объект или нет. Соответствующий процесс ввода получает от устройства ввода точные координаты. Удобен при использовании мыши, светового пера, сенсорного экрана.

При относительном выборе пределы перемещения по экрану ограничены только списком объектов. Пользователь всегда будет выбирать объекты только из этого списка.

Для информировании пользователя о положении указателя выбора используются изменение атрибутов, графические указатели, перемещение курсора и т.п.

При выборе необходимо определить способ завершения процесса ввода, т.е. «фиксации» выбранного объекта. С этой целью часто используется нажатие клавиши <Enter>, двойной щелчок мышью.

Методы разработки гибкого интерфейса.

Гибкость интерфейса заключается в способности приложения адаптироваться (пользователем или автоматически) к любому возможному уровню подготовки пользователя. Существуют три вида адаптации: фиксированная, полная и косметическая.

При *фиксированной адаптации* пользователь явно выбирает уровень диалоговой поддержки. Простейший вариант такой адаптации основан на использовании правила двух уровней, согласно которому система обеспечивает два вида диалога: подробный (для начинающего пользователя); краткий (для подготовленного пользователя).

Правило двух уровней может быть расширено до правила N уровней диалога. Однако такой подход имеет несколько недостатков:

1) не учитывается тот факт, что навыки накапливаются постепенно;
2) пользователь может хорошо знать одну часть системы и совсем не знать другую;
3) пользователь сам определяет уровень своей подготовки, что снижает объективность оценки.

При *полной адаптации* диалоговая система стремится построить модель пользователя, которая по мере обучения последнего и определяет стиль диалога в зависимости от этих изменений. При этом одной из основных проблем является распознавание характеристик пользователя. Для ее решения необходимо определить, что использовать в качестве таких характеристик: время, затрачиваемое пользователем на ответ, количество обращений за помощью или характер ошибок и тип запрашиваемой помощи.

В настоящее время полная (автоматическая) адаптация практически ни в одной диалоговой системе не реализована.

Косметическая адаптация призвана обеспечить гибкость диалога без учета поведения пользователя, но и без однозначного выбора им конкретного стиля диалога. Такая адаптация может быть достигнута за счет применения следующих методов:

- использование умолчаний;
- использование сокращений;
- опережающий ввод ответов;

- многоуровневая помощь;
- многоязычность.

Использование умолчаний. Сущность умолчания состоит в том, что система использует некоторое изначально заданное значение какого-либо параметра, пока пользователь не изменит его. В этом случае имеют место два аспекта адаптации системы: во-первых, начинающий пользователь имеет возможность использовать большинство параметров системы по умолчанию; во-вторых, система может запоминать значения, либо заданные при последнем сеансе работы, либо наиболее часто используемые.

Для удобства начинающих пользователей значения, используемые по умолчанию, могут выводиться на экран вместе с соответствующим вопросом системы, например: «Дата регистрации документа? [текущая]».

Самый распространенный способ принятия значений по умолчанию – это нулевой ввод, т.е. просто нажатие клавиши <Enter> в качестве ответа на вопрос системы. Если используется командный язык, то пользователь просто пропускает параметр, используемый по умолчанию.

Использование сокращений предполагает, что пользователь вместо полного имени команды может вводить ее любое допустимое сокращенное обозначение. На первый взгляд может показаться, что сокращенный ввод более удобен для начинающего пользователя. Но это не совсем так. Чтобы пользователь мог, не задумываясь, заменить команду корректным сокращением, он должен достаточно хорошо представлять имеющийся набор команд, усвоить лексику системы.

Одной из модификаций этого подхода является *опережающий ввод символов*, при котором система, «узнав» по первым символам команду, «дописывает» ее сама.

Идея ***опережающего ввода ответов*** заключается в том, что пользователь имеет возможность на очередном шаге диалога вводить не один ответ, а цепочку последовательных ответов, упреждая возможные вопросы системы.

Один из методов обеспечения ***многоуровневой помощи*** состоит в том, что сначала на экран выводится сообщение начального уровня, а затем пользователь может уточнить полученную информацию, используя переход на более низкий уровень по ключевому слову. На таком принципе основана работа многих современных Help-систем, обучающих гипертекстовых систем.

Сущность ***многоязычности интерфейса*** состоит в том, что структура и семантика диалоговых сообщений, которые выдает и получает пользователь, должны отвечать нормам родного языка пользователя и не зависеть от того, на каком языке разработаны инструментальные средства, которые он использует.

Возможный подход к реализации многоязычности – создание средств реакции системы на действия пользователя (сообщения-запросы, подсказки, сообщения об ошибках) отдельно от синтаксиса языка программирования (инструментальных средств).

Темп ведения диалога.

Темп ведения диалога зависит от характеристик аппаратных и программных средств, а также от специфики решаемых задач. Кроме того, темп ведения диалога зависит от психофизиологических характеристик пользователя.

Время ответа (отклика) системы определяется как интервал между событием и реакцией системы на него. Данная характеристика интерфейса определяет задержку в работе пользователя при переходе к выполнению следующего шага задания.

Требования к времени ответа зависят от того, что ожидает пользователь от работы системы, и от того, как взаимодействие с системой влияет на выполнение заданий. Исследования показали, что если время ответа меньше ожидаемого, точность выбора операции из меню увеличивается с увеличением времени ответа системы. Это связано с тем, что излишне быстрый ответ системы как бы подгоняет пользователя, заставляет его суетиться в стремлении не отставать от более расторопного партнера по общению.

Время ответа должно соответствовать естественному ритму работы пользователя. В обычном разговоре люди ожидают ответа около 2 секунд и ждут того же при работе с компьютером. Время ожидания зависит от их состояния и намерений. На представления пользователя оказывает сильное влияние также его предшествующий опыт работы с системой.

Обычно человек может одновременно запомнить сведения о пяти-девяти предметах. Считается также, что хранение данных в кратковременной памяти ограничено по времени: около 2 с. для речевой информации и 30 с. для сенсорной. Поэтому люди разбивают свою деятельность на этапы, соответствующие порциям информации, которые они могут хранить одновременно в памяти.

Завершение очередного этапа называется ***клаузой***. Задержки, препятствующие наступлению клаузы, очень вредны и неприятны, т.к. содержимое кратковременной памяти требует постоянного обновления и легко стирается под влиянием внешних факторов. Зато после клаузы подобные задержки вполне приемлемы и даже необходимы.

Завершение задачи, ведущее к отдыху, называют ***закрытием***. В этот момент исчезает необходимость дальнейшего хранения информации и человек получает существенное психологическое облегчение.

Так как пользователи интуитивно стремятся к закрытию в своей работе, следует делить диалоги на фрагменты, чтобы пользователь мог вовремя забывать промежуточную информацию.

Пользователи, особенно новички, обычно предпочитают много мелких операций одной большой операции, т.к. в этом случае они могут не только лучше контролировать общее продвижение решения и обеспечить его удовлетворительный ход, но и отвлечься от деталей работы на предыдущих этапах.

Имеющиеся результаты исследований позволили выработать следующие рекомендации по допустимому времени ответа интерактивной системы:

0,1-0,2 с. – для подтверждения физических действий (нажатие клавиши, работа со световым пером, мышью);

0,5-1 с. – для ответа на простые команды (от момента ввода команды, выбора альтернативы из меню до появления нового изображения на экране);

1-2 с. – при ведении связного диалога (когда пользователь воспринимает серию взаимосвязанных вопросов как одну порцию информации для формирования одного или нескольких ответов, задержка между следующими друг за другом вопросами не должна превышать указанную длительность);

2-4 с. – для ответа на сложный запрос, состоящий в заполнении некоторой формы. Если задержка не влияет на другую работу пользователя, связанную с первой, могут быть приемлемы задержки до 10 с.;

более 10 с. – при работе в мультизадачном режиме, когда пользователь воспринимает данную задачу как фоновый процесс. Принято считать, что если пользователь не получает ответ в течение 20 с., то это не интерактивная система. В таком случае пользователь может забыть о задании, заняться решением другой задачи и возвращаться к нему тогда, когда ему будет удобно. При этом программа должна сообщать пользователю, что задержка ответа не является следствием выхода системы из строя (например, путем регулярного обновления строки состояния системы или ведения протокола выполнения задания пользователя).