4.9. Интерфейс пользователя интерфейсы командной строки. Текстовые интерфейсы. Графические интерфейсы (WIMP-интерфейсы), оконные менеджеры. Реализация интерфейсов пользователя.

(UI — англ. *user interface*) — совокупность средств, при помощи которых пользователь общается с различными устройствами, чаще всего — с компьютером или бытовой техникой, либо иным сложным инструментарием (системой).

Интерфейс пользователя компьютерного приложения включает:

- средства отображения информации, отображаемую информацию, форматы и коды;
- командные режимы, язык «пользователь — интерфейс»;
- устройства и технологии ввода данных;
- диалоги, взаимодействие и транзакции между пользователем и компьютером, обратную связь с пользователем;
- поддержку принятия решений в конкретной предметной области;
- порядок использования программы и документацию на неё.

Пользовательский интерфейс часто понимают только как внешний вид программы. Однако на деле пользователь воспринимает через него всю программу в целом, а значит, такое понимание является слишком узким.

В действительности ПИ объединяет в себе все элементы и компоненты программы, которые способны оказывать влияние на взаимодействие пользователя с программным обеспечением (ПО).

Это не только экран, который видит пользователь.

К этим элементам относятся:

- набор задач пользователя, которые он решает при помощи системы;
- используемая системой метафора (например, рабочий стол в MS Windows®);
- элементы управления системой;
- навигация между блоками системы;
- визуальный (и не только) дизайн экранов программы;
- средства отображения информации, отображаемая информация и форматы;
- устройства и технологии ввода данных;
- диалоги, взаимодействие и транзакции между пользователем и компьютером;
- обратная связь с пользователем;
- поддержка принятия решений в конкретной предметной области;
- порядок использования программы и документация на нее.

Для упрощения восприятия функции программы пользователем при разработке пользовательского интерфейса желательно использовать метафоры.

1 Интерфейс командной строки

разновидность текстового интерфейса человека и компьютера, в котором инструкции компьютеру даются только путём ввода с клавиатуры текстовых строк (*команд*). Также известен под названием *консоль*.

Интерфейс командной строки противопоставляется системам управления программой на основе меню, а также различным реализациям графического интерфейса.

Формат вывода информации в интерфейсе командной строки не регламентируется; обычно это также простой текстовый вывод, но может быть и графическим, звуковым и т.д.

4.9

1.1 Назначение

На устройстве-консоли, которое печатало текст на бумаге, интерфейс командной строки был единственным возможным. На видеотерминалах интерфейс командной строки применяется по таким причинам:

- Небольшой расход памяти по сравнению с системой меню.
- В современном программном обеспечении имеется большое число команд, многие из которых нужны крайне редко. Поэтому даже в некоторых программах с графическим интерфейсом применяется командная строка: набор команды (при условии, что пользователь знает эту команду) осуществляется гораздо быстрее, чем, например, навигация по меню.
- Естественное расширение интерфейса командной строки — пакетный интерфейс. Его суть в том, что в файл обычного текстового формата записывается последовательность команд, после чего этот файл можно выполнить в программе, что возымеет такой же (не меньший) эффект, как если бы эти команды были по очереди введены в командную строку.

Если программа полностью или почти полностью может управляться командами интерфейса командной строки и поддерживает пакетный интерфейс, умелое сочетание интерфейса командной строки с графическим предоставляет пользователю очень мощные возможности.

1.2 Формат команды

Наиболее общий формат команд (в квадратные скобки помещены необязательные части):

[символ_начала_команды]имя_команды [параметр_1 [параметр_2 […]]]

Символ начала команды может быть самым разным, однако чаще всего для этой цели используется косая черта (**/**). Если строка вводится без этого символа, выполняется некоторая базовая команда: например, строка «**Привет**» в IRC эквивалентна вводу «**/msg Привет**». Если же такой базовой команды нет, символ начала команды отсутствует вообще (как, например, в DOS).

Параметры команд могут иметь самый разный формат. В основном применяются следующие правила:

- параметры разделяются пробелами (и отделяются от названия команды пробелом)
- параметры, содержащие пробелы, обрамляются кавычками-апострофами (**'**) или двойными кавычками (**"**)
- если параметр используется для обозначения включения какой-либо опции, выключенной по умолчанию, он начинается с косой черты (**/**) или дефиса (**-**)
- если параметр используется для включения/выключения какой-либо опции, он начинается (или заканчивается) знаком плюс или минус (для включения и выключения соответственно)
- если параметр указывает действие из группы действий, назначенных команде, он не начинается со специальных символов
- если параметр указывает объект, к которому применяется действие команды, он не начинается со специальных символов
- если параметр указывает дополнительный параметр какой-либо опции, то он имеет формат **/опция:дополнительный_параметр** (вместо косой черты также может употребляться дефис)

Например, в некоей абстрактной игре может быть такая команда:

```
/map dm1 /skill:2
```

4.9

- `/` — символ начала команды
- **`map`** — название команды (переход на другой уровень)
- **`dm1`** — обязательный параметр (название уровня)
- **`/skill:2`** — дополнительный параметр (задание уровня сложности)

1.3 Применение

Основные сферы применения интерфейса командной строки:

- операционные системы
- чаты
- компьютерные игры

В операционных системах Основное применение интерфейса командной строки — интерфейс операционной системы. В Windows язык командной строки не имеет чёткой стандартизации, однако существует стандарт командной строки POSIX и его модификация в рамках GNU (см. командная оболочка UNIX).

В других программах

- AutoCAD
- различные клиенты IRC

1.4 Достоинства и недостатки

Достоинства:

- Любую команду можно вызвать небольшим количеством нажатий.
- Пакетные файлы — это, по сути, простейшая программируемость.
- Можно управлять программами, не имеющими графического интерфейса (например, выделенным сервером).
- Просмотрев содержимое консоли, можно повторно увидеть промелькнувшее сообщение, которое вы не успели прочитать.

Недостатки:

- Интерфейс командной строки не является дружественным для начинающих.
- Искать неизвестную команду по справочникам не менее сложно, чем отыскивать в меню нужную команду.
- Ввод некоторых параметров с клавиатуры может быть затруднительным. Например, игроки часто украшают свои имена цифрами и спецсимволами, и ввести команду наподобие `kick =-CooL-= [H3LL]` без дополнительных средств бывает довольно сложно. А подбор громкости с помощью озвученного ползунка позволяет выставить подходящую громкость быстрее, чем из командной строки.
- Если же в программе должен быть полноценный скриптовый язык, приходится либо поддерживать два разных языка (консольный и скриптовый), либо отказываться от командной строки в пользу скриптового языка, либо совмещать эти два языка (что отрицательно сказывается на удобстве программирования).

2 Текстовый интерфейс пользователя

4.9

(англ. *Text user interface, TUI*) — система средств взаимодействия пользователя с компьютером, основанная на использовании текстового (буквенно-цифрового) режима дисплея или аналогичных — например, командная строка.

2.1 Особенности текстового интерфейса

На программном уровне, для ввода и вывода информации консольные программы используют *стандартные устройства ввода-вывода* (stdin, stdout, stderr), хотя могут открывать и другие файлы, сетевые соединения и совершать иные действия, доступные в выполняющей их среде. Вывод печатных символов в stdout и stderr приводит к появлению этих символов на устройстве вывода, т.е. к их получению пользователем.

В простейшем случае, консольная программа использует интерфейс командной строки, однако, многие из них, с помощью управляющих последовательностей терминалов, создают более дружественный интерфейс, приближающийся к графическому (см. меню (информатика)).

Некоторые консольные программы пригодны лишь для определённой реализации текстового интерфейса, например текстовые программы DOS. Однако, консольные программы для более продвинутых операционных систем, особенно UNIX, как правило, способны работать на достаточно широком классе реализаций интерфейса с пользователем. Для упрощения написания таких программ широко применяется библиотека ncurses.

2.2 Реализация текстового интерфейса

В принципе, консольная программа не обязана заботиться о реализации самого взаимодействия с пользователем, ограничиваясь вводом-выводом на *стандартные устройства*, использованием библиотек типа ncurses или иных программных интерфейсов. Собственно взаимодействие с пользователем обычно осуществляет операционная система или иное программное обеспечение.

Классической реализацией текстового интерфейса, восходящей к первой половине XX века, является алфавитно-цифровое устройство ввода-вывода, например комплект из клавиатуры и АЦПУ (телетайпа). Впоследствии вместо АЦПУ стали применять мониторы, снабжённые знакогенератором, что позволило быстро и удобно организовывать диалог с пользователем. Подобными устройствами снабжён, или может быть снабжён, почти каждый современный компьютер. Такие комплекты из монитора и клавиатуры (иногда с добавлением мыши) называются консолью компьютера.

В соответствии с традицией использования консольными программами клавиатуры и АЦПУ для ввода и вывода соответственно, взаимодействие таких программ с пользователем свелось к чтению из stdin и выводу на stdout. Таким образом, появилась возможность перенаправлять потоки ввода-вывода, осуществляя взаимодействие с пользователем посредством иных устройств, в т.ч. подключенных через сеть, а также при помощи специальных программ-эмуляторов терминала, например рисующих окно с текстом в графическом интерфейсе пользователя (*текстовое окно*).

В 1970-х гг. и позднее выпускались даже специальные устройства, реализующие текстовый интерфейс — *текстовые терминалы*, подключаемые через последовательный порт к компьютеру, напрямую или через модем. С распространением персональных компьютеров функции текстового терминала как правило выполняет компьютер, тот на котором выполняется консольная программа или другой. Программы Telnet и ssh (а также PuTTY для Windows) позволяют пользователю взаимодействовать с консольной программой, запущенной на удалённом компьютере (как правило под управлением UNIX), через Интернет или локальную сеть. Программы xterm, rxvt и konsole реализуют текстовый интерфейс посредством текстового окна в среде X Window.

Альтернативный подход к консольному выводу был использован в персональных компьютерах, в частности (хотя не только) IBM PC под управлением DOS. Программа может не только выводить данные через stdout, но и прямо изменять содержимое

определённой области памяти, связанной со знакогенератором монитора, приводя к немедленному изменению видимых на мониторе данных. Такие программы могут также работать в среде Microsoft Windows. Более того, Windows имеет поддержку текстовых окон, во многом превосходящую имевшуюся в DOS, в т.ч. и для приложений собственно Windows.

Linux предоставляет ещё бóльшие возможности для консольных программ. В частности, даже безо всякого графического интерфейса несколько одновременно запущенных программ могут бесконфликтно взаимодействовать с пользователем, создавая иллюзию наличия в системе нескольких консолей (виртуальные консоли).

2.3 Примеры консольных программ

- Любая программа, осуществляющая получение данных от пользователя путём чтения stdin и отправку данных пользователю путём записи в stdout, по определению является консольной программой. Однако, такие программы могут обходиться и безо всякого пользователя, например обрабатывая данные из файлов.
- Текстовые программы для DOS, осуществляющие вывод в видеопамять EGA/VGA. Подобные программы работают также в среде GUI Microsoft Windows.
- Unix shell, а также все утилиты, предназначенные для работы в этой среде.
- Midnight Commander (UNIX), FAR Manager (Windows).

3 Графический интерфейс пользователя

(ГИП, англ. *graphical user interface*, GUI) в вычислительной технике — система средств для взаимодействия пользователя с компьютером, основанная на представлении всех доступных пользователю системных объектов и функций в виде графических компонентов экрана (окон, значков, меню, кнопок, списков и т. п.). При этом, в отличие от интерфейса командной строки, пользователь имеет произвольный доступ (с помощью клавиатуры или устройства координатного ввода типа «мышь») ко всем видимым экранным объектам.

Впервые концепция ГИП была предложена учеными из исследовательской лаборатории Xerox PARC в 1970-х, но получила коммерческое воплощение лишь в продуктах корпорации Apple Computer. В операционной системе AmigaOS ГИП с многозадачностью был использован в 1985 г. В настоящее время ГИП является стандартной составляющей большинства доступных на рынке операционных систем и приложений.
Примеры систем, использующих ГИП: Mac OS, Solaris, GNU/Linux, Microsoft Windows, NEXTSTEP, OS/2, BeOS.

4 WIMP

In human–computer interaction, **WIMP** stands for "window, icon, menu, pointing device", denoting a style of interaction using these elements. It was coined by Merzouga Wilberts in 1980. Although its usage has fallen out of favor, it is often used as an approximate synonym of "GUI". WIMP interaction was developed at Xerox PARC (see Xerox Alto, developed in 1973) and "popularized by the Macintosh computer in 1984".[1]

This style of interaction uses a physical input device to control the position of a cursor and presents information organized in windows and represented with icons. Available commands are compiled together in menus and actioned through the pointing device. This reduces the cognitive load to remember the possibilities available, reducing learning times.

4.9

Other benefits of this style include its ease of use for non-technical people, both novice and power users. Also know-how can be ported from one application to the next, given the high consistency between interfaces.

Since "wimp" in common speech is a derogatory term for a person lacking strength or courage, the acronym WIMP is sometimes used in a likewise derogatory manner[1], especially by those who prefer more traditional command-line interfaces.

4.1 Alternative Expansions

Different sources expand the acronym **WIMP** differently. The terms may be plural or singular, and the term corresponding to P varies the most. All of the following can be found on the web (as of 2004):

- W: Window(s)
- I: Icon(s)
- M: Menu(s); Mouse/Mice (rarely) (note that mice are a subset of pointing devices)
- P: Pointing device(s); Pointing; Pointer(s) (note that the term "pointer" is often used as a synonym for mouse cursor); Pull-down menu(s) (note that pull-down menus can be thought of as a subset of menus) It is normally just "Pointer" though

Another possibility is to have the P in WIMP stand for Program, allowing it to be used as a noun (like the noun GUI, for graphical user interface) rather than as an adjective or qualifier.

4.2 Criticism

User interfaces based on the WIMP style are very good at abstracting workspaces, documents, and their actions. Their analogous paradigm to documents as paper sheets or folders, makes WIMP interfaces easy to introduce to novice users. Furthermore their basic representations as rectangular regions on a 2D flat screen make them a good fit for system programmers. Generality makes them very suitable for multitasking work environments.

This explains why the paradigm has been prevalent over more than 20 years, both giving rise to and benefiting from the availability of commercial widget toolkits that support this style. Though several HCI researchers consider this to be a sign of stagnation in user interface design, a current lack of innovation in the search for new interaction models.

There are applications for which WIMP is not well suited, they argue, and the lack of technical support increase difficulty for the development of their interfaces. This include any application requiring devices that provides continuous input signals, showing 3D models, or simply portraying an interaction for which there is not defined any standard widget. Andries van Dam call these interfaces post-WIMP GUIs.

5 Менеджер окон

Философия построения X Window System очень похожа на философию построения UNIX, «инструменты, не политика». Это значит, что X не пытаются диктовать то, как должна быть выполнена работа. Вместо этого пользователю предоставляются инструменты, а за пользователем остается принятие решения о том, как использовать эти инструменты.

Этот подход расширен в X тем, что не задается, как окна должны выглядеть на экране, как их двигать мышью, какие комбинации клавиш должны использоваться для переключения между окнами (то есть Alt+Tab, в случае использования Microsoft Windows), как должны выглядеть заголовки окон, должны ли в них быть кнопки для закрытия, и прочее.

Вместо этого X делегирует ответственность за это приложению, которое называется «Window Manager» (**Менеджер Окон**). Есть десятки оконных менеджеров для X: AfterStep, Blackbox, ctwm, Enlightenment, fvwm, Sawfish, twm, WindowMaker и другие.

4.9

Каждый из этих оконных менеджеров предоставляет различные внешние виды и удобства; некоторые из них поддерживают «виртуальные рабочие столы»; некоторые из них позволяют изменять назначения комбинаций клавиш, используемых для управления рабочим столом; в некоторых есть кнопка «Start» или нечто подобное; некоторые поддерживают «темы», позволяя изменять внешний вид, поменяв тему.

Кроме того, оболочки KDE и GNOME обе имеют собственные оконные менеджеры, которые интегрированы с оболочкой.

Каждый оконный менеджер также имеет собственный механизм настройки; некоторые предполагают наличие вручную созданного конфигурационного файла; некоторые предоставляют графические инструменты для выполнения большинства работ по настройке; по крайней мере один (Sawfish) имеет конфигурационный файл, написанный на диалекте языка Lisp.

5.1 Политика фокусирования

Другой особенностью, за которую отвечает оконный менеджер, является «политика фокусирования указательным устройством». Каждая оконная система должна иметь некоторый способ выбора окна для активации получения нажатий клавиш, а также визуальную индикацию того, какое окно активно.

Широкоизвестная политика фокусировки называется «click-to-focus». Эта модель используется в Microsoft Windows, когда окно становится активным после получения щелчка мыши.

X сама по себе не поддерживает никакой конкретной политики фокусирования. Вместо этого менеджер окон управляет тем, какое окно владеет фокусом в каждый конкретный момент времени. Различные оконные менеджеры поддерживают разные методы фокусирования. Все они поддерживают метод щелчка для фокусирования, и большинство из них поддерживают некоторые другие методы.

Ниже перечислены самые популярные политики фокусирования.

5.2 Focus-follows-mouse (фокус следует за мышью)

Фокусом владеет то окно, что находится под указателем. Это не обязательно будет окно, которое находится поверх всех остальных. Фокус меняется при указании на другое окно, при этом также нет нужды щёлкать на нём.

5.3 Sloppy-focus (нечёткий фокус)

С политикой focus-follows-mouse если указатель находится поверх корневого окна (или заднего фона), то никакое окно фокус не получает, а нажатия клавиш просто пропадают. При использовании политики нечёткого фокуса он меняется только когда указатель попадает на новое окно, но не когда уходит с текущего окна.

5.4 Click-to-focus (щелчок для выбора фокуса)

Активное окно выбирается нажатием кнопки на указательном устройстве. При этом окно «поднимается», и находится поверх всех других обычных окон. Все нажатия клавиш теперь будут направляться в это окно, даже если указатель переместится к другому.

Многие оконные менеджеры поддерживают и другие политики, а также вариации перечисленных.