Міністерство освіти та науки України

Львівський державний інститут новітніх технологій та управління

ім. В. Чорновола

Факультет інформаційних технологій

**Реферат**

На тему: “Проектування інтерфейсу”

Виконав:

Ст. групи КН-42

Попик М. Р

Львів 2008

**Проектування інтерфейсу користувача**

Мета цього розділу - познайомити з основними аспектами проектування інтерфейсу користувача, які повинні знати розробники ПЗ. Вивчивши цей розділ, ми мали б знати наступне:

1. знати основні принципи проектування інтерфейсу користувача;

2. освоїти п'ять різних стилів взаємодії користувача з програмними системами;

3. знати різні стилі представлення інформації і те, в яких випадках доцільне графічне представлення даних;

4. познайомитися з основними правилами проектування засобів підтримки користувача, вбудованих в програмне забезпечення;

5. мати уявлення про основні показники зручності використання систем.

Проектування обчислювальних систем охоплює широкий спектр проектних дій - від проектування апаратних засобів до проектування інтерфейсу користувача. Організації-розробники часто наймають фахівців для проектування апаратних засобів і дуже рідко для проектування інтерфейсів. Таким чином, фахівцям з розробки ПЗ часто доводиться проектувати і інтерфейс користувача. Якщо у великих компаніях до цього процесу залучаються фахівці з інженерної психології, то в невеликих компаніях послугами таких фахівців практично не користуються.

Грамотно спроектований інтерфейс користувача украй важливий для успішної роботи системи. Складний в застосуванні інтерфейс, як мінімум, приводить до помилок користувача. Іноді вони просто відмовляються працювати з програмною системою, не дивлячись на її функціональні можливості. Якщо інформація представляється плутано або непослідовно, користувачі можуть зрозуміти її неправильно, внаслідок чого їх подальші дії можуть привести до пошкодження даних або навіть до збою в роботі системи.

У 1982 році, інтерфейси користувача були текстовими або створювалися у вигляді спеціальних форм. Зараз майже всі користувачі працюють на персональних комп'ютерах. Всі сучасні персональні комп'ютери підтримують графічний інтерфейс користувача (*graphical user interface* - GUI), який має на увазі використання кольорового графічного екрану з високим розширенням і дозволяє працювати з мишею і з клавіатурою.

Хоча текстові інтерфейси ще достатньо широко застосовуються, особливо в успадкованих системах, у наш час користувачі вважають за краще працювати з графічним інтерфейсом. У табл. 15.1 перераховані основні елементи GUI.

**Таблиця 1. Елементи графічних інтерфейсів користувача**

|  |  |
| --- | --- |
| **Елементи** | **Опис** |
| Вікна | Дозволяють відображати на екрані інформацію різного роду |
| Піктограми | Представляють різні типи даних. У одних системах піктограми представляють файли, в інших - процеси |
| Меню | Введення команд замінюється вибором команд з меню |
| Покажчики | Миша використовується як пристрій вказівки для вибору команд з меню і для виділення окремих елементів у вікні |
| Графічні елементи | Можуть використовуватися спільно з текстовими |

***Графічні інтерфейси володіють рядом переваг:***

1. Їх відносно просто вивчити і використовувати. Користувачі, що не мають досвіду роботи з комп'ютером, можуть легко і швидко навчитися працювати з графічним інтерфейсом.

2. Кожна програма виконується в своєму вікні (екрані). Можна перемикатися з однієї програми в іншу, не втрачаючи при цьому дані, отримані в ході виконання програм.

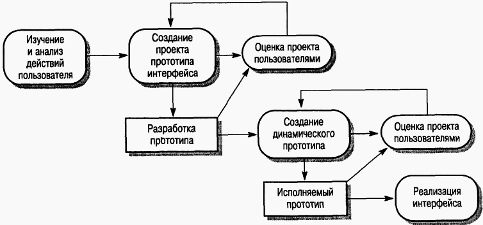
3. Режим повноекранного відображення вікон дає можливість прямого доступу до будь-якого місця екрану.

Мета даного розділу - привернути увагу розробників ПЗ до деяких ключових проблем, лежачих в основі проектування інтерфейсів користувача. Розробники і програмісти зазвичай компетентні у використанні таких технологій, як класи Swing в мові Java [103] або HTML [249], реалізації інтерфейсів користувача, що є основою. Проте цю технологію далеко не завжди застосовують належним чином, внаслідок чого інтерфейси користувача виходять неелегантними, незручними і складними у використанні.

У цьому розділі я приведу декілька рекомендацій по проектуванню засобів кінцевого користувача, не розглядаючи весь процес проектування цих засобів. Із-за браку місця розглядаються тільки графічні інтерфейси. Спеціальні інтерфейси, наприклад для мобільних телефонів, телевізійних приймачів, копіювальної техніки або факсимільних апаратів, розглядатися не будуть. Тут я зроблю тільки коротке введення в тему проектування інтерфейсів користувача.

На мал. 15.1 зображений ітераційний процес проектування призначеного для користувача інтерфейсу. Найбільш ефективним підходом до проектування інтерфейсу користувача є розробка із застосуванням моделювання призначених для користувача функцій. На початку процесу створюються паперові макети інтерфейсу, потім розробляються екранні форми, що моделюють взаємодію з користувачем. Бажано, щоб кінцеві користувачі брали активну участь в процесі проектування інтерфейсу .

У одних випадках користувачі допоможуть оцінити інтерфейс; у інших будуть повноправними членами проектної групи.



*Мал. 15,1. Процес проектування інтерфейсу користувача*

Важливим етапом процесу проектування інтерфейсу користувача є аналіз діяльності користувачів, яку повинна забезпечити обчислювальна система. Не вивчивши того, що, з погляду користувача, повинна робити система, неможливо сформувати реалістичний погляд на проектування ефективного інтерфейсу. Для аналізу потрібно (як правило, одночасно) застосовувати різні методики, а саме: аналіз завдань, етнографічний підхід , опити користувачів і спостереження за їх роботою.

**2. Принципи проектування інтерфейсів користувача**

Розробники інтерфейсів завжди повинні враховувати фізичні і розумові здібності людей, які працюватимуть з програмним забезпеченням. Люди на короткий час можуть запам'ятати вельми обмежений об'єм інформації і здійснюють помилки, якщо доводиться вводити уручну великі об'єми даних або працювати в напружених умовах. Фізичні можливості людей можуть істотно розрізнятися, тому при проектуванні інтерфейсів користувача необхідно постійно пам'ятати про це.

Основою принципів проектування інтерфейсів користувача є людські можливості. У табл. 15.2 представлені основні принципи, застосовні при проектуванні будь-яких інтерфейсів користувача.

**Таблиця 2. Принципи проектування інтерфейсів користувача**

|  |  |
| --- | --- |
| **Принцип** | **Опис** |
| Облік знань користувача | У інтерфейсі необхідно використовувати терміни і поняття, узяті з досвіду майбутніх користувачів системи. |
| Узгодженість | Інтерфейс повинен бути узгодженим в тому сенсі, що однотипні (але різні) операції повинні виконуватися одним і тим же способом. |
| Мінімум несподіванок | Поведінка системи повинна бути прогнозованою. |
| Здатність до відновлення | Інтерфейс повинен мати засоби, що дозволяють користувачам відновити дані після помилкових дій. |
| Керівництво користувача | Інтерфейс повинен надавати необхідну інформацію у разі помилок користувача і підтримувати засоби контекстно-залежної довідки. |
| Облік різнорідності користувачів | У інтерфейсі повинні бути засоби для зручної взаємодії з користувачами, що мають різний рівень кваліфікації і різні можливості. |

*Принцип обліку знань користувача припускає наступне*: інтерфейс повинен бути настільки зручний при реалізації, щоб користувачам не знадобилося особливих зусиль, щоб звикнути до нього. У інтерфейсі повинні використовуватися терміни, зрозумілі користувачеві, а об'єкти, керовані системою, повинні бути безпосередньо пов'язані з робочим середовищем користувача. Наприклад, якщо розробляється система, призначена для авіадиспетчерів, то керованими об'єктами в ній повинні бути літаки, траєкторії польотів, сигнальні знаки і тому подібне . Основну реалізацію інтерфейсу в термінах файлових структур і структур даних необхідно приховати від кінцевого користувача. Принцип узгодженості інтерфейсу користувача припускає, що команди і меню системи повинні бути одного формату, параметри повинні передаватися у всі команди однаково і пунктуація команд повинна бути схожою. Такі інтерфейси скорочують час на навчання користувачів. Знання, отримані при вивченні якої-небудь команди або частини додатку, можна потім застосувати при роботі з іншими частинами системи.

В даному випадку мова йде про узгодженості низького рівня. І творці інтерфейсу завжди повинні прагнути до нього. Проте бажана узгодженість і більш високого рівня. Наприклад, зручно, коли для всіх типів об'єктів системи підтримуються однакові методи (такі, як друк, копіювання і тому подібне). Проте повна узгодженість неможлива і навіть небажана. Наприклад, операцію видалення об'єктів робочого столу доцільно реалізувати за допомогою їх перетягання в корзину. Але в текстовому редакторові такий спосіб видалення фрагментів тексту здається неприродним.

Завжди потрібно дотримувати наступний принцип: кількість несподіванок повинна бути мінімальною, оскільки користувачів дратує, коли система раптом починає поводитися непередбачувано. При роботі з системою у користувачів формується певна модель її функціонування. Якщо його дія в одній ситуації викликає певну реакцію системи, природно чекати, що таке ж дія в іншій ситуації приведе до аналогічної реакції. Якщо ж відбувається зовсім не те, що очікувалося, користувач або дивується, або не знає, що робити. Тому розробники інтерфейсів повинні гарантувати, що схожі дії справлять схоже враження.

Дуже важливий принцип відновлюваності системи, оскільки користувачі завжди допускають помилки. Правильно спроектований інтерфейс може зменшити кількість помилок користувача (наприклад, використання меню дозволяє уникнути помилок, які виникають при введенні команд з клавіатури), проте всі помилки усунути неможливо. У інтерфейсах повинні бути засоби, що по можливості запобігають помилкам користувача, а також що дозволяють коректно відновити інформацію після помилок. Ці засоби бувають двох видів.

1. *Підтвердження деструктивних дій -* Якщо користувач вибрав потенційно деструктивну операцію, то він повинен ще раз підтвердити свій намір.

2. *Можливість відміни дій -* Відміна дії повертає систему в той стан, в якому вона знаходилася до їх виконання. Не зайвою буде підтримка багаторівневої відміни дій, оскільки користувачі не завжди відразу розуміють, що зробили помилку.

Наступний принцип - підтримка користувача. Засоби підтримки користувачів повинні бути вбудовані в інтерфейс і систему і забезпечувати різні рівні допомоги і довідкової інформації. Повинне бути декілька рівнів довідкової інформації - від основ для початкуючих до повного опису можливостей системи. Довідкова система повинна бути структурованою і не перенавантажувати користувача зайвою інформацією при простих запитах до неї .

Принцип обліку різнорідності користувачів припускає, що з системою можуть працювати різні їх типи. Частина користувачів працює з системою нерегулярно, час від часу. Але існує і інший тип "досвідчені користувачі", які працюють з додатком щодня по декілька годин. Випадкові користувачі потребують такого інтерфейсу, який "керував" би їх роботою з системою, тоді як досвідченим користувачам потрібний інтерфейс, який дозволив би їм максимально швидко взаємодіяти з системою. Крім того, оскільки деякі користувачі можуть мати різні фізичні недоліки, в інтерфейсі повинні бути засоби, які допомогли б їм перенастроювати інтерфейс під себе. Це можуть бути засоби, що дозволяють відображати збільшений текст, заміщати звук текстом, створювати кнопки великих розмірів і тому подібне

Принцип визнання різноманіття категорій користувачів може суперечити іншим принципам проектування інтерфейсів, наприклад узгодженості інтерфейсу. Аналогічно, необхідний рівень довідкової інформації для різних типів користувачів може радикально відрізнятися. Неможливо створити таку довідкову систему, яка підійшла б всім користувачам. Розробник інтерфейсу повинен завжди бути готовим до компромісних рішень залежно від реальних користувачів системи.

**3. Взаємодія з користувачем**

Розробникові інтерфейсу користувача обчислювальних систем необхідно вирішити два головні завдання: яким чином користувач вводитиме дані в систему і як дані будуть представлені користувачеві. "Правильний" інтерфейс повинен забезпечувати і взаємодію з користувачем, і представлення інформації.

Інтерфейс користувача забезпечує введення команд і даних в обчислювальну систему. На перших обчислювальних машинах був тільки один спосіб введення даних - через інтерфейс командного рядка, причому для взаємодії з машиною використовувалася спеціальна командна мова. Такий спосіб годився тільки для досвідчених користувачів, тому пізніше були розроблені спрощені способи введення даних. Всі ці види взаємодії можна віднести до одного з п'яти основних стилів взаємодії :

1. *Безпосереднє маніпулювання.* Користувач взаємодіє з об'єктами на екрані. Наприклад, для видалення файлу користувач просто перетягує його в корзину.

2. *Вибір з меню.* Користувач вибирає команду із списку пунктів меню. Дуже часто вибрана команда впливає тільки на той об'єкт, який виділений (вибраний) на екрані. При такому підході для видалення файлу користувач спочатку вибирає файл, а потім команду на видалення.

3. *Заповнення форм.* Користувач заповнює поля екранної форми. Деякі поля можуть мати своє меню (випадне меню або списки). У формі можуть бути командні кнопки, при клацанні мишею на яких ініціюють деяку дію. Щоб видалити файл за допомогою інтерфейсу, заснованого на формі, треба ввести в поле форми ім'я файлу і потім клацнути на кнопці видалення, присутній у формі.

4. *Командна мова.* Користувач вводить конкретну команду з параметрами, щоб вказати системі, що вона повинна далі робити. Щоб видалити файл, користувач вводить команду видалення з ім'ям файлу як параметр цієї команди.

5. *Природна мова.* Користувач вводить команду на природній мові. Щоб видалити файл, користувач може ввести команду "Видалити файл з ім'ям XXX".

Кожен з цих стилів взаємодії має переваги і недоліки і найкращим чином підходить різним типам додатків і різним категоріям користувачів. У табл. 15.3 перераховані основні переваги і недоліки перерахованих стилів взаємодії і вказані типи додатків, в яких вони зазвичай використовуються.

Звичайно, стилі взаємодії рідко використовуються в чистому вигляді, в одному застосуванні може використовуватися одночасно декілька різних стилів. Наприклад, в операційній системі Microsoft Window підтримується декілька стилів: пряме маніпулювання піктограмами, що представляють файли і теки, вибір команд з меню, ручне введення деяких команд, таких як команди конфігурації системи, використання форм (діалогових вікон).

**Таблиця 3.**

**Переваги і недоліки стилів взаємодії користувача з системою**

|  |  |  |  |
| --- | --- | --- | --- |
| **Стиль взаємодії** | **Основні переваги** | **Основні недоліки** | **Приклади додатків** |
| Пряме маніпулювання | Швидке і інтуїтивно зрозуміле взаємодія. Легкий у вивченні | Складна реалізація. Підходить тільки там, де є зоровий образ завдань і об'єктів | Відеоігри; системи автоматичного проектування |
| Вибір з меню | Скорочення кількості помилок користувача.  Введення з клавіатури мінімальне | Повільний варіант для досвідчених користувачів. Може бути складним, якщо меню складається з великої кількості вкладених пунктів | Головним чином системи загального призначення |
| Заповнення форм | Простій введення даних.  Легкий у вивченні | Займає простір на екрані | Системи управління запасами; обробка фінансової інформації |
| Командна мова | Могутній і гнучкий | Важкий у вивченні. Складно запобігти помилкам введення | Операційні системи; бібліотечні системи |
| Природна мова | Підходить недосвідченим користувачам.  Легко настроюється | Вимагає великого ручного набору | Системи розкладу; системи зберігання даних WWW |

Призначені для користувача інтерфейси додатків World Wide Web базуються на засобах, що надаються мовою HTML (мова розмітки Web-сторінок) разом з іншими мовами, наприклад Java, який пов'язує програми з компонентами Web-сторінок. В основному інтерфейси Web-сторінок проектуються для випадкових користувачів і є інтерфейсами у вигляді форм. У Web-додатках можна створювати інтерфейси, в яких застосовувався б стиль прямого маніпулювання, проте до моменту написання книги проектування таких інтерфейсів представляло достатньо складне в аспекті програмування завдання.

В принципі необхідно застосовувати різні стилі взаємодії для управління різними системними об'єктами. Даний принцип складає основу моделі Сихейма (Seeheim) призначених для користувача інтерфейсів .У цій моделі розділяються представлення інформації, управління діалоговими засобами і управління додатком. Насправді така модель є швидше ідеальною, чим практичною, проте майже завжди є можливість розділити інтерфейси для різних класів користувачів (наприклад, початкуючих і досвідчених). На мал. 15.2 зображена подібна модель з розділеними інтерфейсом командної мови і графічним інтерфейсом, лежача в основі деяких операційних систем, зокрема Linux.



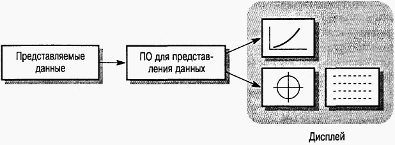
*Мал. 2. Множинний інтерфейс*

Розділення уявлення, взаємодії і об'єктів, включених в інтерфейс користувача, є основним принципом підходу "модель-представлення-контролер", який обговорюється в наступному розділі. Ця модель порівнянна з моделлю Сихейма, проте використовується при реалізації окремих об'єктів інтерфейсу, а не всього застосування.

**4. Представлення інформації**

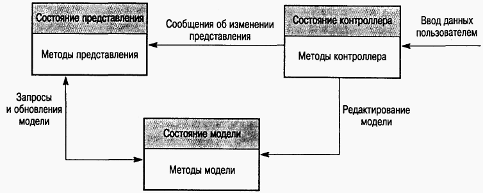
У будь-якій інтерактивній системі повинні бути засоби для представлення даних користувачам. Дані в системі можуть відображатися по-різному: наприклад, інформація, що вводиться, може відображатися безпосередньо на дисплеї (як, скажімо, текст в текстовому редакторові) або перетворюватися в графічну форму. Хорошим тоном при проектуванні систем вважається відділення представлення даних від самих даних. До деякої міри розробка такого ПЗ суперечить об'єктно-орієнтованому підходу, при якому методи, що виконуються над даними, повинні бути визначені самими даними. Проте в нашім випадку передбачається, що розробник об'єктів завжди знає якнайкращий спосіб представлення даних; хоча це, звичайно, не завжди так. Часто визначити якнайкращий спосіб представлення даних конкретного типу досить важко, у такому разі об'єктні структури не повинні бути "жорсткими".

Після того, як представлення даних в системі відокремлене від самих даних, зміни в уявленні даних на екрані користувача відбуваються без зміни самої системи (мал. 15.3).



*Мал. 3. Представлення даних*

Підхід "модель-представление-контроллер" (МПК), представлений на мал..4, отримав первинне застосування в мові Smalltalk як ефективний спосіб підтримки різних представлень даних. Користувач може взаємодіяти з кожним типом уявлення. Дані, що відображаються, інкапсульовані в об'єкти моделі. Кожен об'єкт моделі може мати декілька окремих об'єктів уявлень, де кожне уявлення - це різні відображення моделі.



*Мал. 4. Модель МПК взаємодії з користувачем*

Кожне уявлення має пов'язаний з ним об'єкт контролера, який обробляє введені користувачем дані і забезпечує взаємодію з пристроями. Така модель може представити числові дані, наприклад, у вигляді діаграм або таблиць. Модель можна редагувати, змінюючи значення в таблиці або параметри діаграми.

Щоб знайти якнайкраще представлення інформації, необхідно знати, з якими даними працюють користувачі і яким чином вони застосовуються в системі. Ухвалюючи рішення за уявленням даних, розробник повинен враховувати ряд чинників.

1. Що потрібне користувачеві - точні значення даних або співвідношення між значеннями?

2. Наскільки швидко відбуватимуться зміни значень даних? Чи потрібно негайно показувати користувачеві зміну значень?

3. Чи повинен користувач робити які-небудь дії у відповідь на зміну даних?

4. Чи потрібно користувачеві взаємодіяти з інформацією, що відображається, за допомогою інтерфейсу з прямим маніпулюванням?

5. Інформація повинна відображатися в текстовому (описово) або числовому форматі? Чи важливі відносні значення елементів даних?

Якщо дані не змінюються протягом сеансу роботи з системою, їх можна представити або в графічному, або в текстовому вигляді, залежно від типу додатку. Текстове представлення даних займає на екрані мало місця, але у такому разі дані не можна охопити одним поглядом. За допомогою різних стилів уявлення незмінні дані слід відокремити від даних, що динамічно змінюються. Наприклад, статичні дані можна виділити особливим шрифтом, підкреслити особливим кольором або позначити піктограмами.

Якщо потрібна точна цифрова інформація і дані змінюються відносно поволі, їх можна відображати в текстовому вигляді. Там, де дані змінюються швидко, зазвичай використовується графічне уявлення.

Як приклад розглянемо систему, яка щомісячно записує і підбиває підсумки за даними продажів якійсь компанії. На мал. 15.5 видно, що одні і ті ж дані можна представити у вигляді тексту і в графічному вигляді.

Менеджерам, що вивчають дані про продажі, зазвичай більше потрібні тенденції зміни або аномальні дані, чим їх точні значення. Графічне представлення цієї інформації у вигляді гістограми дозволяє виділити аномальні дані за березень і травень, що значно відрізняються від решти даних. Як видно з мал. 15.5 дані в текстовому уявленні займають менше місця, чим в графічному.

Динамічні зміни числових даних краще відображати графічно, використовуючи аналогові уявлення. Цифрові екрани, що постійно змінюються, збивають користувачів з пантелику, оскільки точні значення даних швидко не сприймаються. Графічне відображення даних при необхідності можна доповнити точними значеннями. Різні способи представлення числових даних, що змінюються, показані на мал. 15.6.

Безперервні аналогові відображення допомагають спостерігачеві оцінити відносні значення даних. На мал. 15.7 числових значень температури і тиску приблизно однакові. Але при графічному відображенні видно, що значення температури близьке до максимального, тоді як значення тиску не досягло навіть 25% від максимуму. Зазвичай, окрім поточного значення, спостерігачеві потрібно знати максимальні (або мінімальні) можливі значення. Він винен в думці обчислювати відносний стан прочитуваних даних. Додатковий час, необхідний для розрахунків, може привести до помилок оператора в стресових ситуаціях, коли виникають проблеми і на дисплеї відображаються аномальні дані.

При представленні точних буквено-цифрових даних для виділення особливій інформації можна використовувати графічні елементи. Замість звичайного рядка дані краще помістити в прямокутник або відзначити піктограмою (мал. 15.8). Прямокутник з повідомленням поміщається поверх поточного екрану, тим самим привертаючи до нього увагу користувача.

Виділення інформації за допомогою графічних елементів можна також використовувати для залучення уваги до змін, що відбуваються в різних частинах екрану. Але, якщо зміни відбуваються дуже швидко, не слід використовувати графічні елементи, оскільки швидкі зміни можуть привести до накладення екранів, що збиває з пантелику і дратує користувачів.

При представленні великих об'ємів даних можна використовувати різні прийоми візуалізація, яка указує на споріднені елементи даних. Розробники інтерфейсів повинні пам'ятати про можливості візуалізацію, особливо якщо інтерфейс системи повинен відображати фізичну суть (об'єкти). Ось декілька *прикладів візуалізації даних.*

1. Відображення метеорологічних даних, зібраних з різних джерел, у вигляді метеорологічних карт з ізобарами, повітряними фронтами і тому подібне

2. Графічне відображення стану телефонної мережі у вигляді зв'язаної безлічі вузлів.

3. Візуалізація стану хімічного процесу з показом тиску і температур в групі зв'язаних між собою резервуарів і труб.

4. Модель молекули і маніпулювання нею в тривимірному просторі за допомогою системи віртуальної реальності.

5. Відображення безлічі Web-сторінок у вигляді дерева гіпертекстових посилань .

**4.1. Використання в інтерфейсах кольору**

У всіх інтерактивних системах, незалежно від їх призначення, підтримуються кольорові екрани, тому в призначених для користувача інтерфейсах часто використовуються різні кольори. У деяких системах кольори застосовують в основному для виділення певних елементів (наприклад, в текстових редакторах для виділення фрагментів тексту); у інших системах (таких, як системи автоматичного проектування) кольори позначають різні рівні проектів.

Правильне використання кольорів робить інтерфейс користувача зручнішим для розуміння і управління. Разом з тим використання кольорів може бути неправильним, внаслідок чого створюються інтерфейси, які візуально непривабливі і навіть провокують помилки. Основним принципом розробників інтерфейсів повинне бути обережне використання кольорів  
 на екранах. У роботі дається 14 правил ефективного використання кольору в призначених для користувача інтерфейсах. Ось найбільш важливі з них.

1. *Використовуйте обмежену кількість кольорів.* Для вікон не слід використовувати більше чотирьох або п'яти різних кольорів, в інтерфейсі системи не повинно бути більше семи кольорів.

2. *Використовуйте різні кольори для показу змін в стані системи.* Якщо на екрані змінилися кольори, значить, відбулася якась подія. Виділення кольором особливо важливе в складних екранах, в яких відображаються сотні різних об'єктів.

3. *Для допомоги користувачеві використовуйте колірне кодування.* Якщо користувачам необхідно виділяти аномальні елементи, виділите їх кольором; якщо потрібно знайти подібні елементи, виділите їх однаковим кольором.

4. *Використовуйте колірне кодування продумано і послідовно.* Якщо в якій-небудь частині системи повідомлення про помилку відображаються, наприклад, червоним кольором, то у всіх інших частинах подібні повідомлення повинні відображатися таким же кольором. Тоді червоний колір не слід використовувати де-небудь ще. Якщо ж червоний колір використовується ще десь в системі, користувач може інтерпретувати появу червоного кольору як повідомлення про помилку. Слід пам'ятати, що у певних типів користувачів є свої уявлення про значення окремих кольорів.

5. *Обережно використовуйте доповнюючі кольори.* Фізіологічні особливості людського ока не дозволяють одночасно сфокусуватися на червоному і синьому кольорах. Тому послідовність червоних і синіх зображень викликає зорова напруга. Деякі комбінації кольорів також можуть візуально порушувати або утрудняти читання.

Найчастіше розробники інтерфейсів допускають дві помилки: прив'язка значення до певного кольору і використання великої кількості кольорів на екрані. Використовувати кольори для представлення значення не слід по двох причинах. Близько 10% людей мають нечітке уявлення про кольори і тому можуть неправильно інтерпретувати значення. У різних груп людей різне сприйняття кольорів; крім того, в різних професіях існують свої угоди про значення окремих кольорів. Користувачі на підставі отриманих знань можуть неадекватно інтерпретувати один і той же колір. Наприклад, водієм червоний колір сприймається як *небезпека.* А у хіміка червоний колір означає *гарячий.*

При використанні дуже яскравих кольорів або дуже великої їх кількості відображення стають плутаними. Різноманіття кольорів збиває з пантелику користувача (так, наприклад, на деякі абстрактні картини не можна дивитися тривалий час без напруги) і викликає у нього зорове стомлення. Непослідовне використання кольорів також дезорієнтує користувача.

**5. Засоби підтримки користувача**

У першому розділі цього розділу був запропонований принцип проектування, згідно якому інтерфейс користувача повинен завжди забезпечувати деякий тип оперативної довідкової системи. Довідкові системи - один з основних аспектів проектування інтерфейсу користувача. Довідкову систему додатку складають:

• повідомлення, що генеруються системою у відповідь на дії користувача;

• діалогова довідкова система;

• документація, що поставляється з системою.

Оскільки проектування корисної і змістовної інформації для користувача - справа вельми серйозне, воно повинне оцінюватися на тому ж рівні, що і архітектура системи або програмний код. Проектування повідомлень вимагає значного часу і чималих зусиль. Доречно привертати до цього процесу професійних письменників і художників-графіків. При проектуванні повідомлень про помилки або текстової довідки необхідно враховувати чинники, перераховані в табл. 15.4

**Таблиця 5. Чинники проектування текстових повідомлень**

|  |  |
| --- | --- |
| **Чинник** | **Опис** |
| Зміст | Довідкова система повинна знати, що робить користувач, і реагувати на його дії повідомленнями відповідного змісту |
| Досвід користувача | Якщо користувачі добре знайомі з системою, їм не потрібні довгі і докладні повідомлення. В той же час початкуючим користувачам такі повідомлення здадуться складними, малозрозумілими і дуже короткими. У довідковій системі повинні підтримуватися обидва типи повідомлень, а також повинні бути засоби, що дозволяють користувачеві управляти складністю повідомлень |
| Професійний рівень користувача | Повідомлення повинні містити відомості, відповідні професійному рівню користувачів. У повідомленнях для користувачів різного рівня необхідно застосовувати різну термінологію |
| Стиль повідомлень | Повідомлення повинні мати позитивний, а не негативний відтінок. Завжди слід використовувати активний, а не пасивний тон звернення. У повідомленнях не повинно бути образ або спроб пожартувати |
| Культура | Розробник повідомлень повинен бути знайомий з культурою тієї країни, де продається система. Повідомлення, цілком доречне в культурі однієї країни, може виявитися неприйнятним в іншій |

**5.1. Повідомлення про помилки**

Перше враження, яке користувач отримує при роботі з програмною системою, грунтується на повідомленнях про помилки. Недосвідчені користувачі, зробивши помилку, повинні зрозуміти повідомлення, що з'явилося, про помилку.

Новачки і досвідчені користувачі повинні передбачати ситуації, при яких можуть виникнути повідомлення про помилки. Наприклад, хай користувачем системи є медсестра госпіталю, що працює у відділенні інтенсивної терапії. Обстеження пацієнтів виконується на відповідному устаткуванні, пов'язаному з обчислювальною системою. Щоб проглянути поточний стан пацієнта, користувач системи вибирає пункт меню Показати і набирає ім'я пацієнта в полі введення (мал. 15.9).

Хай медсестра ввела ім'я пацієнта Bates, замість Pates. Система не знаходить пацієнта з таким ім'ям і генерує повідомлення про помилку. Повідомлення про помилку повинні бути завжди ввічливими, короткими, послідовними і конструктивними, не містити образ. Не слід також використовувати звукові сигнали або інші звуки, які можуть збити з пантелику користувача. Непогано включити в повідомлення варіанти виправлення помилки. Повідомлення про помилку повинне бути пов'язане з контекстно-залежною довідкою.

На мал. 15.10 показані приклади двох повідомлень про помилку. Повідомлення, розташоване зліва, спроектоване погано. Воно негативне (звинувачує користувача в здійсненні помилки), не адаптоване до рівня знань і досвідченості користувача, не враховує змісту помилки. У цьому повідомленні не пропонуються способи виправлення ситуації, що склалася. Крім того, в повідомленні використані специфічні терміни (номер помилки), не зрозумілі користувачеві. Повідомлення справа набагато краще. Воно позитивне, в нім використовуються медичні терміни і пропонується простій спосіб виправлення помилки за допомогою клацання на одній з кнопок. У разі потреби користувач може викликати довідку.

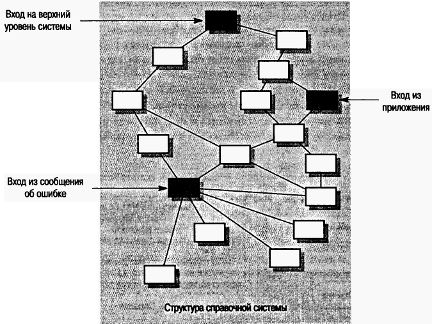
**5.2. Проектування довідкової системи**

При отриманні повідомлення про помилку користувач часто не знає, що робити, і звертається до довідкової системи за інформацією. Довідкова система повинна надавати різні типи інформації: як ту, що допомагає користувачеві в скрутних ситуаціях, так і конкретну інформацію, яку шукає користувач. Для цього довідкова система повинна мати різні засоби і різні структури повідомлень.

Довідкова система повинна забезпечувати користувачеві декілька різних точок входу (мал. 15.11). Користувач може увійти до неї на верхньому рівні її ієрархічної структури і тут оглянути всі розділи довідкової інформації. Інші точки входу в довідкову систему - за допомогою вікон повідомлень про помилки або шляхом отримання опису конкретної команди додатку.

Всі довідкові системи мають складну мережеву структуру, в якій кожен розділ довідкової інформації може посилатися на дещо інших інформаційних розділів. Структура такої мережі, як правило, ієрархічна, з перехресними посиланнями, як показано на мал. 15.11. Нагорі структурної ієрархії міститься загальна інформація, внизу - докладніша.

При використанні довідкової системи виникають проблеми, пов'язані з тим, що користувачі входять в мережу після здійснення помилки і потім переміщаються по мережі довідкової системи. Через деякий час вони заплутуються і не можуть визначити, в якому місці довідкової системи знаходяться. У такій ситуації користувачі повинні завершити сеанс роботи з довідковою системою і знов почати роботу з деякої відомої точки довідкової мережі.



*Мал. 15.11. Точки входу в довідкову систему*

Відображення довідкової інформації в декількох вікнах спрощує подібну ситуацію. На мал. 15.12 показаний екран, на якому розташовано три вікна довідки. Проте простір екрану завжди обмежений і розробникові слід пам'ятати, що додаткові вікна можуть приховати іншу потрібну інформацію.

Тексти довідкової системи необхідно готувати спільно з розробниками додатку. Довідкові розділи не повинні бути просто відтворенням керівництва користувача, оскільки інформація на папері і на екрані сприймається по-різному. Сам текст (а також його розташування і стиль) повинен бути ретельно продуманий, щоб його можна було прочитати у вікні щодо малого розміру. Розділ довідки **Переадресація** пошти на мал. 15.12 порівняно невеликий - в будь-якому довідковому розділі повинна бути тільки найнеобхідніша інформація. У вікні, що відображає довідковий розділ, розташовано три кнопки: для показу додатковій інформації, для переміщення по тексту розділу і для виклику списку довідкових тим.

У вікні **Журнал\*** показаний список вже проглянутих розділів. Можна повернутися в один з них, вибравши відповідний пункт із списку. Вікно навігації по довідковій системі - це графічна "карта" мережі довідкової системи. Поточна позиція на карті повинна бути виділена кольором, тінями або, як в нашому випадку, підписом.

У користувачів є декілька можливостей переміщення між розділами довідкової системи: можна перейти до розділу безпосередньо з розділу, що відображається, можна вибрати потрібний розділ з вікна **Журнал**, щоб проглянути його ще раз, і, нарешті, можна вибрати відповідний вузол на карті довідкової мережі і перейти до цього вузла.

Довідкову систему можна реалізувати у вигляді групи зв'язаних Web-страниц або за допомогою узагальненої гіпертекстової системи, інтегрованої з додатком. Ієрархічна структура легко реалізується у вигляді гіпертекстових посилань. Web-системы мають переваги: вони прості в реалізації і не вимагають спеціального програмного забезпечення. Проте при створенні контекстно-залежної довідки можуть виникнути труднощі при пов'язанні її з додатком.

**5.3. Документація користувача**

Строго кажучи, документація не є частиною призначеного для користувача інтерфейсу, проте проектування оперативної довідкової підтримки разом з документацією є хорошим правилом. Системне керівництво надає докладнішу інформацію, ніж діалогові довідкові системи, і будуються так, щоб бути корисними користувачам різного рівня.

Для того, щоб документація, що поставляється спільно з програмною системою, була корисна всім системним користувачам, вона повинна містити п'ять описаних нижче документів (або, можливо, розділів в одному документі) (мал. 15.13).

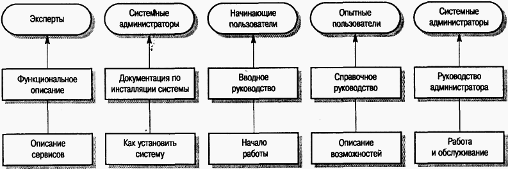
1. *Функціональний опис,* в якому стисло представлені функціональні можливості системи. Прочитавши функціональний опис і ввідне керівництво, користувач повинен визначити, чи та це система, яка йому потрібна.

2. *Документ по інсталяції системи, в* якому міститься інформація по установці системи. Тут повинні бути відомості про диски, на яких поставляється система, опис файлів, що знаходяться на цих дисках, і мінімальні вимоги до конфігурації. У документі повинні бути інструкція по інсталяції і докладніша інформація по установці файлів, залежних від конфігурації системи.

3. *Ввідне керівництво,* що представляє неформальне введення в систему, що описує її "повсякденне" використання. У цьому документі повинна міститися інформація про те, як почати роботу з системою, як використовувати загальні можливості системи. Всі описи повинні бути забезпечені прикладами і містити відомості про те, як відновити систему після помилки і як почати наново роботу. У книзі [68] запропонований ефективний спосіб складання ввідного керівництва, при якому основна увага приділена відновленню системи після помилок, а решта всієї інформації, необхідної користувачам, зводиться до мінімуму.

4. *Довідкове керівництво,* в якому описані можливості системи і їх використання, представлений список повідомлень про помилки і можливі причини їх появи, розглянуті способи відновлення системи після виявлення помилок.

5. *Керівництво адміністратора,* необхідне для деяких типів програмних систем. У нім даний опис повідомлень, що генеруються системою при взаємодії з іншими системами, і описані способи реагування на ці повідомлення. Якщо в систему включена апаратна частина, то в керівництві адміністратора повинна бути інформація про те, як виявити і усунути несправності, пов'язані з апаратурою, як підключити нові периферійні пристрої і тому подібне



*Мал. 13. Типи призначеної для користувача документації*

Разом з перерахованим керівництвом необхідно надавати іншу зручну в роботі документацію. Для досвідчених користувачів системи зручні різного вигляду наочні покажчики, які допомагають швидко проглянути список можливостей системи і способи їх використання.

**5.4. Оцінювання інтерфейсу**

Це процес, в якому оцінюється зручність використання інтерфейсу і ступінь його відповідності вимогам користувача. Таким чином, оцінювання інтерфейсу є частиною загального процесу тестування і атестації систем ПІВ

У ідеалі оцінювання повинне проводитися відповідно до показників зручності використання інтерфейсу, перерахованих в табл. 15.5. Кожен з цих показників можна оцінити чисельно. Наприклад, вивченість можна оцінити таким чином: досвідчений оператор після тригодинного навчання повинен уміти використовувати 80% функціональних можливостей системи. Проте частіше зручність використання інтерфейсу оцінюється якісно, а не через числові показники.

**Таблиця 5. Показники зручності використання інтерфейсу**

|  |  |
| --- | --- |
| **Показник** | **Опис** |
| Ізучаємость | Кількість часу навчання, необхідне для початку продуктивної роботи з системою |
| Швидкість роботи | Швидкість реакції системи на дії користувача |
| Стійкість | Стійкість системи до помилок користувача |
| Відновлюваність | Здатність системи відновлюватися після помилок користувача |
| Адаптується | Здатність системи "підстроюватися" до різних стилів роботи користувачів |

Повне оцінювання призначеного для користувача інтерфейсу може виявитися вельми дорогим, в цей процес будуть залучені фахівці з когнітивної психології і дизайнери. У процес оцінювання можуть входити розробка і виконання ряду статистичних експериментів з користувачами в спеціально створених лабораторіях і з необхідним для спостереження устаткуванням. Таке оцінювання інтерфейсу економічно нерентабельно для систем, що розробляються в невеликих організаціях з обмеженими ресурсами.

Існують простіші і менш дорогі методики оцінювання інтерфейсів користувача, що дозволяють виявити окремі дефекти в інтерфейсах.

1. Анкети, в яких користувач дає оцінку інтерфейсу.

2. Спостереження за роботою користувачів з подальшим обговоренням їх способів використання системи при вирішенні конкретних завдань.

3. Відеоспостереження типового використання системи.

4. Додавання в систему програмної коди, яка збирала б інформацію про найчастіше використовувані системні сервіси і найбільш поширені помилки.

Анкетування користувачів - відносно дешевий спосіб оцінки інтерфейсу. Питання повинні бути точними, а не загальними. Не слід використовувати питання типу "Будь ласка, прокоментуйте практичність системи", оскільки відповіді, ймовірно, істотно розрізнятимуться. Краще ставити конкретні питання, наприклад: "Оціните зрозумілість повідомлень про помилки за шкалою від 1 до 5. Оцінка 1 означає повністю зрозуміле повідомлення, 5 - малозрозуміле". На такі питання легко відповісти і ймовірніше отримати в результаті корисну для поліпшення інтерфейсу інформацію.

Під час заповнення анкети користувачі повинні обов'язково оцінити власний досвід і знання. Такого роду відомості дозволять розробникам зафіксувати, користувачі з яким рівнем знань мають проблеми з інтерфейсом. Якщо проект інтерфейсу вже створений і пройшов оцінювання в паперовому вигляді, анкети можна використовувати навіть до повної реалізації системи.

При спостереженні користувачів за роботою оцінюється, як вони взаємодіють з системою, які використовують сервіси, які здійснюють помилки і тому подібне Разом із спостереженнями можуть проводитися семінари, на яких користувачі розповідають про свої спроби вирішити ті або інші проблеми і про те, як вони розуміють систему і як використовують її для досягнення цілей.

Відеоустаткування відносне недорого, тому до безпосереднього спостереження можна додати відеозапис призначених для користувача семінарів для подальшого аналізу. Повний аналіз відеоматеріалів дорогий і вимагає спеціально оснащеного комплекту з декількома камерами, направленими на користувача і на екран. Проте відеозапис окремих дій користувача може виявитися корисним для виявлення проблем. Щоб визначити, які саме дії викликають проблеми у користувача, слід удатися до інших методів оцінювання.

Аналіз відеозаписів дозволяє розробникові встановити, чи багато рухів руками вимушений здійснювати користувач (у деяких системах користувачеві постійно доводиться переходити з клавіатури на мишу), і виявити неприродні рухи очей. Якщо при роботі з інтерфейсом потрібно часто зміщувати зоровий фокус, користувач може зробити більше помилок і пропустити які-небудь частини зображення.

Вставка в програму коди, що збирає статистичні дані при використанні системи, покращує інтерфейс декількома способами. Виявляються найбільш часто використовувані операції. Інтерфейс змінюється так, щоб ці операції вибиралися швидше в порівнянні з іншими. Наприклад, у вертикальному або випадному меню найбільш часто використовувані команди повинні знаходитися вверху списку. Такий код також дозволить виявити і змінити команди, сприяючі появі помилок.

Нарешті, в кожній програмі повинні бути нескладні засоби, за допомогою яких користувач зможе передавати розробникам повідомлення з "скаргами". Такі засоби переконують користувачів в тому, що на їх думку зважають. А розробники інтерфейсу і інші фахівці можуть отримати швидкий зворотний зв'язок щодо окремих проблем інтерфейсу.

Жоден з цих далеко не складних методів оцінки призначеного для користувача інтерфейсу не є надійним і не гарантує вирішення всіх проблем інтерфейсу. Разом з тим перед випуском системи ці методи можна застосувати в групі добровольців, не витрачаючи значних засобів. При цьому виявляється і виправляється більшість проблем в інтерфейсі користувача.

**КЛЮЧОВІ ПОНЯТТЯ**

• Процес проектування інтерфейсу повинен орієнтуватися на користувача. Інтерфейс повинен взаємодіяти з користувачем на його «мові», бути логічним і послідовним. У інтерфейсі повинні бути довідкові засоби, що допомагають користувачам при роботі з системою, і засоби відновлення після помилок.

• Існує декілька стилів взаємодії з програмними системами: безпосереднє маніпулювання, системне меню, заповнення форми, командні мови і природна мова.

• Для відображення тенденцій числових даних і їх приблизних значень слід використовувати графічні уявлення. Числове уявлення повинне застосовуватися тільки тоді, коли потрібно відобразити точні значення даних.

• Кольори в інтерфейсі користувача повинні використовуватися обережно і послідовно. Розробники повинні завжди пам'ятати, що багато хто не розрізняє квітів.

• Повідомлення про помилки не повинні містити звинувачень в адресу користувача. Вони повинні пропонувати варіанти виправлення помилки і забезпечувати зв'язок з довідковою системою.

• У документації користувача повинне бути керівництво для початкуючих і досвідчених користувачів. Для системного адміністратора повинні бути окремі документи.

• У системній специфікації бажано мати кількісні значення для показників зручності використання інтерфейсу, а процес його оцінювання повинен перевіряти систему на відповідність цим вимогам.