

Debreceni Egyetem
Informatikai Kar

Felhasználói felületek ergonómiája

Témavezető:
Dr. Juhász István
Egyetemi adjunktus

Készítette:
Bolyós Lilla
Programtervező Informatikus

Debrecen
2011.

Tartalom

<u>Bevezetés.....</u>	<u>4</u>
<u>1. Szoftver ergonómiai szemléletmód</u>	<u>5</u>
<u>2.Használhatóság, használhatóság összetevői</u>	<u>8</u>
<u>2.1 Használhatóság és hasznosság</u>	<u>8</u>
<u>2.2 Tetszetősség és használhatóság.....</u>	<u>9</u>
<u>2.3 Feltárás, tanulás és hatékonyság</u>	<u>9</u>
<u>2.4 Költségek</u>	<u>12</u>
<u>3.Szoftvertervezés elvei ergonómiai szempontból</u>	<u>14</u>
<u>4.A feladat előzetes megfogalmazása.....</u>	<u>19</u>
<u>4.1 használhatósági módszerek</u>	<u>19</u>
<u>4.2 Feladat előzetes megfogalmazása, feladatot meghatározó mondat.....</u>	<u>22</u>
<u>4.3 A feladatot meghatározó mondat kifejtése.....</u>	<u>25</u>
<u>4.4 Célkitűzések</u>	<u>27</u>
<u>4.5 A követelmények elemzése.....</u>	<u>29</u>
<u>4.6 Használhatósági célok a feladatok előzetes megfogalmazásakor</u>	<u>30</u>
<u>4.6.1 Használhatósági terv</u>	<u>31</u>
<u>5.Felhasználók megismerése, módszerei.....</u>	<u>32</u>
<u>5.1 Vizsgálati módszerek</u>	<u>32</u>
<u>5.1.1 A kérdőíves módszer</u>	<u>36</u>
<u>5.1.2 Interjú.....</u>	<u>39</u>
<u>5.1.3 Személyes beszélgetés</u>	<u>40</u>
<u>5.1.4 Megfigyelési módszerek</u>	<u>41</u>
<u>5.1.5 Szerepátvétel.....</u>	<u>42</u>
<u>6. Kezelőfelület tervezése, interakció stílusai</u>	<u>43</u>
<u>6.1 Az interakció stílusai.....</u>	<u>43</u>
<u>6.2 Billentyűzetet használó, módváltó stílusok.....</u>	<u>44</u>
<u>6.2.1 Menü alapú interakció</u>	<u>44</u>
<u>6.2.2 Kérdés-felelet interakció.....</u>	<u>51</u>

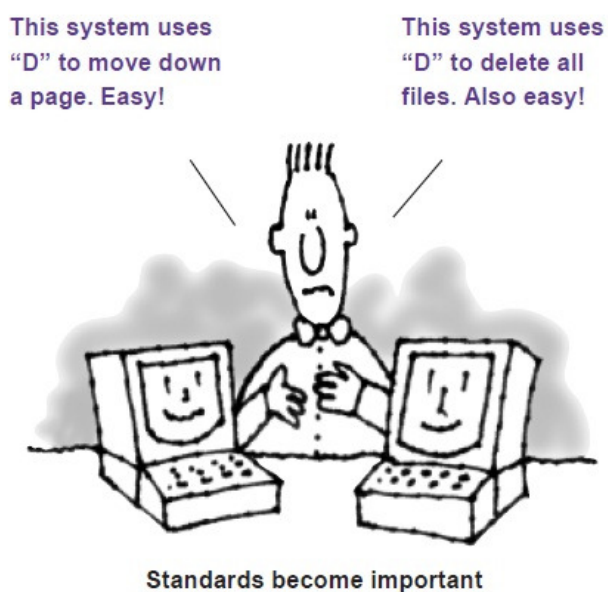
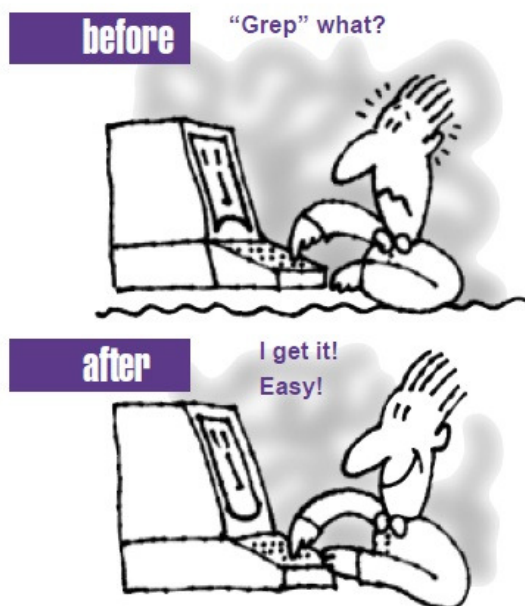
6.2.3 Egyedi kezelő szervekkel irányított interakció.....	52
6.3 Közvetlen manipulációs stílusok	52
6.3.1 Grafikus közvetlen manipuláció	52
6.3.2 Űrlapkitöltés	54
6.3.2.1 Tervezési szempontok	55
6.4 Nyelv alapú stílusok.....	57
6.4.1 Parancsnyelvű interakció	57
6.4.2 Természetes nyelvi interakció	58
7. Munkakörnyezet ergonómiája	59
Befejezés.....	61
Irodalomjegyzék	63

Bevezetés

A felhasználói felületek ergonómiai vizsgálata már a korai szoftverfejlesztőket is foglalkoztatta. A fejlesztők az 1965ös évektől azon fáradoztak, hogy a programok minél jobban használhatóak, precízek és könnyen tanulhatóak legyenek.

Régen nem álltak még rendelkezésünkre grafikus fejlesztői eszközök, a programok parancssoros használta egy átlagos felhasználó számára rendkívüli nehézséget okozott. Ahogy a számítástechnika és az eszközök fejlődtek, úgy váltak jobbra a programok felületei, egyre nagyobb hatékonyságot, egyre jobb tanulhatóságot biztosítva a használóiknak.

Eleinte a legfőbb fejlesztői szempont a jó használhatóság volt, az egyéb igények csak a későbbi években jelentkeztek.



Az 1985ös évektől a fejlesztők és használók rájöttek, hogy a jó használhatóság csak az egyik alapeleme a szoftver ergonómiájának. Fontosnak látták bevezetni a sztenderdeket. Ez nem jelent mást, mint a felület következetes felépítését. Ha az egyik programban a jól megszokott ctrl+c a kimásolás billentyűkombinációja, akkor elvárható, hogy ez a kombináció egy

másik programban is ezt az utasítást jelentse.

Ezen igények felmerülése egy új fejlesztői szemléletmódot kívánt. A szoftver fejlesztési folyamatán belül helyet kell, hogy kapjon a program felületének megtervezése. A felület tervezésének eszközeit tárgyalja ez a dolgozat, útmutatást nyújtva azoknak, akik érdeklődnek a téma iránt, és átfogó képet szeretnének kapni az eszközökről, módszerekről, az összegyűjtött legjobb gyakorlatokról és ezek megvalósítási módjairól.

1. Szoftver ergonómiai szemléletmód

„A technika az emberért van – nem az ember van a technikáért.”

(D. Norman)

Egy programozó számára a szoftver-ergonómiai szemléletmód kialakítása nem minden esetben egyszerű feladat. A programozó a specifikáció alapján elkészíti a szoftvert, beépíti az egyes funkciókat, és csak nagyon ritkán ügyel arra, hogy a kifejlesztett programot nem ő fogja használni, hanem esetleg egy olyan ember, aki keveset ért a számítógéphez. Az esetek többségében a felhasználók nem rendelkeznek informatikai képzettséggel, számukra könnyen érthető és könnyen kezelhető felületeket kell készíteni. Hiába van egy szoftvernek számos hasznos funkciója, ha a kezelése nem egyértelmű, funkciók rejtve maradnak a felhasználók előtt, pedig éppen azért készültek, hogy az ő munkájukat könnyítsék.

A tervezés célja az ember igényeihez való alkalmazkodás, ezért minden vizsgálat az emberből indul ki.

Ahhoz, hogy a felhasználók az elkészült szoftvert a leghatékonyabban tudják használni, célszerű őket bevonni a felület megtervezésébe. Egy felhasználók igényeihez alakított felület megtervezése után a programozónak nem okoz gondot az, hogy a lekódolt funkciókat mely ikonok megnyomásával, vagy esetleg billentyűkombinációk segítségével teszi elérhetővé, vagy milyen struktúrába rendezi a menüket.

Az ember központú tervezés (User-Centered Design, UCD) eredményeként „... az emberek a kívánt műveleteket, szolgáltatásokat, és támogató feladatokat minimális stressz mellett maximális hatékonysággal végezhetik el”.

Az ember központú tervezés három alapelve:

- kiindulás és állandó fókusz: a használó személye és feladatai;
- iteratív tervezés: a tervezés, megvalósítás, tesztelés, és módosítás ismételt alkalmazásával;
- A kezelőfelület minőségét a szoftver-használat kísérleti vizsgálatával (mérésével) ellenőrizhetjük.

Vegyünk példának egy egyszerű alkalmazást, a számológépet.

Tudnunk kell, hogy kik fogják használni. Biztosan van egy réteg, akik csak az alpműveleteket használják (összeadás, kivonás, szorzás, osztás esetleg gyökvonás, négyzetre emelés). Felhasználók egy másik rétege komplexebb feladatok megoldására akarja használni a számológépet. (számrendszerek változtatása, szinusz-koszinusz műveletek).

Ezen információk ismeretében érdemes két felületet tervezni a két felhasználó-réteg számára. Ezen felületeket könnyen kiválaszthatóvá tehetjük menüből, vagy rádiógombok segítségével.

Ez a kis egyszerű alkalmazás is kínálhat több módszert a számok bevitelére (kattintás a felületen, billentyűzet segítségével), így mindenki ki tudja választani a számára kényelmesebbet és praktikusabbat.

Fontos az is, hogy a felhasználó állandó visszajelzést kapjon a felületen arról, hogy sikeresen beírta-e a számokat. Fontos továbbá, hogy ha a felhasználó a felület gombjainak lenyomásával használja a programot, akkor a gombon látszódjon, hogy sikeres volt a kattintás (nyomás animálása, színének megváltozása).

A felületnek kommunikálnia kell a felhasználóval. Például, ha nullával való osztást kísérel meg valaki, akkor egy felugró ablakban, vagy a beviteli mező alatti részen jelezni kell a hibát, és esetleges megoldást kínálni annak javítására. Ilyen esetekben a program hangjelzést is adhat ki, ami szintén figyelemfelkeltő lehet, és ösztönösen az ember rápillant arra, amit eddig beírt és láthatja a hiba okát. Attól függően, hogy mennyire hozzáértő embereknek írtuk a programot, a visszajelző üzeneteket is olyan részletességgel kell megírunk. Például feltehetően, ha egy matematikát tanuló egyetemista próbálná $\tan(90)$ -et kiszámolni, és a program visszajelez neki: hibás input!, minden bizonnyal a fejéhez kap, hogy $\tan(90)$ nem értelmezett. Viszont egy matematikához kevésbé értő számára egy ilyen hibaüzenet sokkal hasznosabb lehet: hibás input, a tangens 90 foknál nem értelmezhető.

Látható, hogy már egy ilyen kis alkalmazás felületének megtervezésekor is sok apróságra oda kell figyelni, egy nagy, összetett alkalmazás esetén még több dolgot kell összehangolni a felhasználók és a felület között.

Ahhoz, hogy egy meghatározott alkalmazási terület szakemberi számára szoftvert tudjunk tervezni és készíteni, meg kell ismernünk azon terület fogalmait, szóhasználatát,

tapasztalatát, az emberek gyakorlottságát. A szoftver-tervező hajlamos az alkalmazási területen dolgozó embereket esetleges hiányos számítástechnikai tudásuk miatt alábecsülni, ez elég nagy hiba. A számítástechnikailag képzetlen használó a maga területén fölöttünk állhat. Egyik fő irányelvünknek az „*Ismerd meg (és tartsd tiszteletben) a használót!*” kell lennie.

Ismert módszere magának a szoftvernek tervezési folyamatnak a végfelhasználókkal készített interjú, valamint a felhasználók napi munkájának megismerése, munkájuk figyelemmel kísérése. Ezen információk segítségével fényt lehet deríteni, hogy melyik funkciókat milyen gyakorisággal használják, hová hányszor kattintanak, jól érzik-e magukat munkavégzés közben. Ha a mindennapi munka során használt alkalmazás olyan, hogy a használata bonyolult, a felhasználók nem ismerik ki magukat a felületen, keresgélniük kell a funkciókat, akkor a munkavégzés lelassul, a hatékonyság romlása figyelhető meg. Mindez stresszhelyzeteket szül, rossz hangulatot a munkahelyen, minden továbbgyűrűzik a dolgozók magánéletébe is.

Tehát egy jól megtervezett felület sokban segít elkerülni a stresszt, szerethetővé teszi a mindennapi rutin munkát, jobb légkör alakulhat ki a munkahelyen.

Sajnos nem minden területen tapasztalható a szoftverek megjelenésének (és egyáltalán a számítógép megjelenésének) pozitív hatásai. Sok helyen (kisebb településeken leginkább) például az orvosi ellátásban még mindig a kézi „papíros” módszert alkalmazzák. Inkább jellemző ma már, hogy a régi módszert párhuzamosan használják a számítógépes programmal. Személyes tapasztalatom az, hogy az asszisztensek egy kartont 1 perc alatt megírnak, de ugyanez a művelet a számítógépen akár 5 percet is igénybe vehet. Vajon annak a szoftvernek a felülete, amelyen ő betölti az adatokat a programba megfelelően van kialakítva? Vajon asszisztensek és orvosok bevonásával tervezték meg a felületet? Ezen kérdésekre véleményem szerint a válasz: nem. Minden bizonnyal, ha az alkalmazás felülete hasonló lenne a papír karton felületéhez, akkor sokkal otthonosabban mozognának az asszisztensek a programban, és rövidebb idő alatt, magabiztosabban töltenék ki a mezőket.

A fenti példák alapján látható, hogy egy jól megtervezett alkalmazás és felülete, mennyiben tudja gyorsítani és segíteni a munkát.

2. Használhatóság, használhatóság összetevői

A használhatóság felhasználói szempontból a (szoftver) rendszerek egy tulajdonsága, hogy mennyire könnyű használni, és mennyire felel meg az azt használó embereknek. A használhatóság fejlesztői szempontból szemlélet és módszere, az ilyen tulajdonságú szoftverek létrehozásában.

A használhatóság fogalma szoftverkészítési kontextusban egy olyan pluszt jelent, ami a felhasználót egyenesen a folyamatok középpontjába helyezi. Ezt a filozófiát felhasználó központú tervezésnek nevezik, ami a tervezési folyamatok elejétől a felhasználók szükségleteit helyezi előtérbe bármilyen tervezési dologról legyen is szó.

Ennek a megközelítésnek a legfontosabb szempontja a használhatóság tesztelés, ami azt jelenti, hogy a felhasználók megosztják a tapasztalataikat, amiket a munkájuk során gyűjtöttek, beszámolnak a termék felületének jóságáról, arról, hogy milyen problémákba ütköztek. Mindezen információkat megosztják a tervezőkkel és a fejlesztőkkel a jobb eredmény elérésének érdekében.

A használhatóság egy olyan mérőeszköz, amellyel megállapítható, hogy milyen egyszerű a termék használata a teljesítendő feladatok tükrében. Ezt mindenképpen meg kell különböztetnünk a termék hasznosságától.

2.1 Használhatóság és hasznosság

A központi tulajdonság, ami meghatározza a termék elfogadhatóságát nem más, mint a hasznavehetősége, ami méri, hogy vajon a termék valós használatával elérhetőek-e a kitűzött célok, amiket a tervezők szándékoztak elérni.

A hasznavehetőség fogalmát két részre bonthatjuk: használhatóság és hasznosság. Habár, ezen fogalmak kapcsolatban állnak egymással, ennek ellenére nem összekeverendőek. A hasznosság a termék problémamegoldó képességére utal. Minél több funkció van egy szoftver termékbe beépítve, annál nagyobb a hasznossága. A használhatóság segít meghatározni, milyen egyszerűen tudnak a felhasználók különböző feladatokat megoldani.

Mindkettő jó minősége szükséges ahhoz, hogy a piacon a termék eladható legyen. Nyilvánvalóan, ha egy program nagyon jól használható, de nem csinál semmi értékeset, értékelhetőt, az embereknek nem lesz oka rá, hogy használják. Azok a felhasználók, akik egy sok funkciót tartalmazó programmal rendelkeznek, amit nehéz használni, akkor minden bizonnyal alternatív megoldást fognak keresni a problémájuk megoldására.

2.2 Tetszetősség és használhatóság

A tetszetősség mindig fontos jellemzője a terméknek. Ha az embereknek tetszik a termék, akkor sokkal szívesebben használják, mint más termékeket. Viszont hasonlóképpen, mint a hasznossággal, a tetszetősséggel is óvatosságnak kell lenni, és vigyázni, hogy ne keverjük össze a tetszetősséget a használhatósággal.

Az emberek sokszor a hasznossággal és használhatósággal összefüggésbe nem hozható okok miatt kedvelnek egy terméket. Talán lebilincseli őket a tervezése, elvakítja őket a „fény és csillogás”. Az emberek hajlamosak a több funkciót tartalmazó termékeket kedvelni, de nem szabad azt gondolnunk, hogy ezek a preferált termékek használhatóbbak.

Leginkább szabványosított kérdőívek használatával mérhetjük azt, hogy mi alapján részesítik előnyben az emberek az egyes termékeket.

2.3 Feltárás, tanulás és hatékonyság

Sok nézőpontja van a használhatóságnak, de az általános fogalmak a feltárás, tanulhatóság és a hatékonyság köré csoportosíthatók.

A feltárás magába foglalja a különleges szükségletek felkutatását, megtalálását, és a termék tulajdonságainak módosítását a felmerült igények tükrében. A használhatóság-tervezés meghatározza, hogy mennyi időbe telik egy felhasználónak, hogy ráleljen az új tulajdonságra, és hogy mennyi hibát vét, amíg bejárja ezt az utat.

A tanulás visszautal arra a felhasználói folyamatra, amikor a felhasználó kitalálja, hogy hogyan használja a felfedezett funkciót ahhoz, hogy teljesíteni tudja a feladatát. Ebben az esetben is a használhatóság-tervezés meghatározza, az idő és a hibaszám arányát, míg a felhasználó megtanulja kezelni az adott funkciót.

A hatékonyság arra a pontra utal, amikor a felhasználó már hatékonyan tudja kezelni a funkciót további tanulás nélkül. A használhatóság-tesztelés jelen esetben meghatározza, hogy egy gyakorlott felhasználó mennyi idő alatt tudja végrehajtani azokat a lépéseket, amik a funkció használatához szükségesek.

A használhatóság három szempontjára nagyban hatással van az, hogy az adott funkció mennyire gyakran van használva, mennyire összetett. Ezekhez az összetett, vagy gyakran használt programrészekhez gyakran készítenek „varázsló” programot, amely segítségével könnyebben elsajátítható a funkció használata.

Nem elegendő csupán hangoztatnunk, hogy a programunk tervezése során alkalmazzuk az ergonómia szabályait, valóban alkalmaznunk kell.

Lehet egy szoftver tökéletesen megtervezve a funkcióit illetően, működhet minden funkciója kifogástalanul, mégis előfordulhat, hogy egy konkurens programot választunk helyette. Ennek oka az lehet, hogy a másik szoftver tervezésekor figyeltek arra is, hogy a felhasználók szempontjából minél könnyebben tanulható, jobban használható legyen. Az együttműködés a felhasználó központú tervezés irányelveivel, sokkal fejlettebb terméket eredményezhet számos területen.

Saját, illetve a szoftverünk támogatását végző emberek erőforrásait is nagyban kímélhetjük, ha betartjuk a szoftver ergonómiai tervezés szabályait. Köztudott, hogy a legtöbb hívás a felhasználóktól a rossz használhatóság miatt van. A felhasználó támogatás viszont nagyon költséges a szoftvert igénybe vevőknek. Ha a felhasználók sokszor falakba ütköznek a használat során, akkor nagyban csökken a felhasználói elégedettség. Ha a program használatát könnyűnek találják a használók, akkor sokkal kevesebbszer hívják a támogatókat.

A másik fontos oka, hogy képezze részét a szoftverfejlesztési folyamatnak a használhatóság az oktatási költségek csökkentése. A jobban használható termék oktatása sokkal rövidebb ideig tart és sokkal hatékonyabb. A felhasználók könnyebben sajátítják el a funkciók használatát.

A használhatósági tesztelés segíti a felhasználói elégedettség növelését. Az elégedettség magában foglalja a következő tényezőket:

- használhatóság
- hasznosság
- tetszetősség

Sokszor a jól használhatóságnak köszönhetően maradnak hűségesek a vevők egy-egy szoftverhez, vagy ajánlják másnak is a terméket. Valamint a hűség következhet abból is, hogy a vevő cég termelékenységé is nőtt a jó használhatóság miatt.

Könnyebb használhatóság → Jobb produktivitás → Nagyobb bevétel a cégnek

A használhatóság segíthet megkülönböztetni a terméket a többitől. Ha két termék közel azonosan hasznos, akkor biztosan a szerint fog a vásárló dönteni közöttük, hogy melyik jobban használható. Apró dolgok is nagy hatással lehetnek a vevők döntésére.

Nem szabad megfeledkeznünk arról, hogy mielőtt a piacra dobjuk a termékünket teszteljük a használhatóságát. Ezzel a lépéssel biztosíthatjuk magunkat, hogy pozitív visszajelzéseket fogunk kapni a felhasználóktól a használhatósággal és a termék egészével kapcsolatban.

2.4 Költségek

A szoftverfejlesztők és projekt menedzserek gyakran aggódnak amiatt, hogy a felhasználó-központú tervezés és a használhatóság tervezés mennyire költséges folyamatok. Elfogadhatatlannak tartják az erre fordított időt és pénzt. Valójában azonban kisebb költséggel lehet számolni akkor, ha már a fejlesztés elejétől bevonjuk a használhatóság tervezést és tesztelést a tervezési folyamatba. Ugyanis, ha ezek figyelembe vétele nélkül fejlesztjük a szoftvert, akkor a kész szoftver felhasználók által való tesztelése után rengeteg negatív visszajelzés érkezhethet hozzánk. Köztük olyanok is, amelyeket feltétlenül orvosolni kell esetleges újratervezéssel. Ennek költsége jóval magasabbra rúghat, mintha kezdetektől bevontuk volna a felhasználókat a tervezésbe.

Ezt olvasva egy szoftverfejlesztő azt gondolhatja, hogy igenis jó dolog tanulni egy kicsit a hasznosságról, hiszen jobbá teheti fele a saját maga fejlesztett szoftvereket. Viszont gyakran szem elől tévesztik a lényegét: a hasznosság és az ügyfél központú tervezés az egész szoftvertervezési folyamaton keresztül kell, hogy íveljen és nem csak néhány lépésben kell odafigyelnünk rá.

A fejlesztőknek fel kell ismerniük azt, hogy ők maguk nem tipikus felhasználók. Nekik sokkal nagyobb rálátásuk van a fejlesztett rendszerre, sokkal nagyobb, mint a felhasználóknak valaha is lesz. Meg kell érteniük, hogy a tesztelés sokkal hatékonyabb az átlagos felhasználók segítségével.

Minél több szoftver fejlesztésekor építjük be a használhatóság elemeit a folyamatokba, annál több tapasztalatunk lesz ebben a témában. Célszerű ezeket összegyűjteni, írásban rögzíteni. Az így összegyűjtött irat együttest használhatósági labornak nevezzük.

Ezek lehetnek

- dokumentumok
- videó felvételek
- hangfelvételek

- interjú anyagok
- tervezési minták.

Ha megszületik az elhatározás, hogy mostantól a használhatóságot is beépítjük a tervezési folyamatokba, akkor két lehetőségünk van:

- magunk alakítjuk ki a tervezési folyamatokat
- szakértők segítségét kérjük

A szakértők többféle segítséget nyújthatnak nekünk:

- segíthetnek felállítani a saját használhatósági laboratóriumunkat
- a tesztelési folyamat során nyújtanak segítséget

Vannak olyan szervezetek, akik segítenek olyan szakembereket találni, akik nagy szakértelemmel bírnak a használhatóság területén, és akik segíthetnek a tervezési folyamatok során.

Egy dologról nem szabad megfeledkezni: akármelyik lehetőséget is választjuk, mindenképpen költséges lesz eleinte a megvalósítás. A szakértőknek fizetnünk kell a tapasztalatukért és a segítségükért. Ők megmondják, mit csinálunk rosszul, de ennek ára van. Úgy kell tekintenünk ezekre a kiadásokra, mint befektetésekre a későbbi nagyobb bevétel reményében.

3. Szofvertervezés elvei ergonómiai szempontból

A tervezési-vizsgálati szempontokat a feladatmegoldás egyes lépéseinél kell figyelembe venni (illetve a vizsgálatoknál számon kérni). Itteni elveink annyiban általánosabbak, hogy ezek a programok egészére, fölépítésére vonatkoznak. Gyakran egyik vagy másik műveletnél vesszük észre, hogy valamelyik elv sérül, de ilyenkor az okokat keresve általában mélyebbre kell ásunk.

Az áttekinthető részek elve:

- alakítsunk ki átlátható, zárt részfeladatokat.

Ez azt jelenti, hogy definiálnunk kell az egyes részfeladatok elejét és végét, hogy tudhassuk, milyen folyamatok alkotják az adott részfeladatot.

A feladatok lezárásának elve:

- A szoftver segítse elő – szöveggel, képpel, zenével – hogy a használóban tudatosodjon egy-egy rész befejezése. (megelégedettség, stb.)

A visszajelzés nagyon fontos a felhasználó számára, a nélkül elveszettnek érzi magát a program használata közben, vagy az adott részfeladat végrehajtása alatt.

A használó kezdeményezésének elve

- Az interakciót az ember irányíthassa. Az ember számára megalázó, ha egy gép diktálja a teendőket. (Emberektől se nagyon szeretjük.) Elfogadjuk fokozottan veszélyes helyzetekben (például szoftver telepítése, egy reaktor beindítása), vagy ha ez nekünk érezhetően segít.

Viszont, mint már hangsúlyoztam, a gép van az emberért és nem az ember a gépért. Általános esetben egy funkció elindítása a felhasználó döntése legyen. Csak abban az esetben vegye át a program az irányítást, ha a felhasználó elrontott valamit. Ebben az esetben is ajánljon fel javítási lehetőségeket, de legalább azt, hogy kéri-e a felhasználó az adott művelet által okozott hiba javítását vagy sem.

Strukturális helyett funkcionális (operatív) dokumentáció

A dokumentáció (szöveges, vagy sűgő) ne azt írja le, hogy hogyan épűl fűl a szoftver, hanem azt, hogy mit lehet vele elvégezni, és hogyan.

Fontos továbbá az is, hogy hozzon példát a sűgő a feladatok megoldására. Bonyolult rendszereknél a lépésről-lépésre történő bemutatás nagyon hasznos lehet, növeli a felhasználói elégedettséget, ha a felhasználó a sűgő segítségével, de mégis egyedűl végre tudja hajtani a bonyolult feladatot.

Egyöntetűség (konzisztencia), ajánlások és szabványok követése

Ugyanazt, vagy hasonlót ugyanűgy, vagy hasonlóan, különbözűt különbözű módon - jelöljűnk, nevezzűnk. Kövessűk a platform szokásait.

Ez azt jelenti, hogy például az az ikon, amelyiken egy kis olló van, mindig jelentse a kivágás funkciűt, akkor is ha a program több felülettal rendelkezik. Nagyon jó példa erre az SAP felhasználói felűlete. Az SAP tervezűi követték a konzisztencia elvét, és minden modulban ugyanazt vagy hasonlűt ugyanűgy, a különbözűt különbözű módon jelölik.

A rendszer állapotának láthatósága

Állandúan látható tájékoztatás a munka állásáról, mindenkor visszajelzés. Az előrehaladás jelzésének elve: az ember (idűnként) nézhesse meg, hogy mi van már készen, és mi van még hátra

Ha egy adott munkafolyamat 10 lépésből áll, akkor legalább számozással szolgáltassunk informáciűt a használnak, hogy hányadik folyamatot hajtja éppen végre. Ez a szám jól látható helyen legyen, ha lehet, akkor többszűr is szerepeljen a felűleten. Jó példa erre egy program telepítése, ami modulokból áll. Vagy gondoljűnk csak arra, amikor az operációs rendszerűnk az automatikus frissítéseket végzi. Mindig látható, hogy hány függű folyamatból, mennyi van még hátra.

A rendszer használata legyen könnyen megismerhető

Egy rendszer megismerése több módon történhet. A felhasználó használhatja a termék súgóját, mint kiindulási alapot. Kérheti a kollégái segítségét, ha elakad egy folyamat végrehajtása közben. Tarthatnak a rendszer fejlesztői tréningeket, amely alkalmakkor a felhasználók elsajátíthatják a program használatát. Ez nagy rendszerek esetében sok időt is igénybe vehet és elég sok pénzbe is kerül. Ezért is fontos törekedni arra, hogy a legkisebb idő- és energiaráfordítással lehessen megismerni a rendszer használatát.

A rendszer illeszkedjék az alkalmazás világához

Számítástechnikai fogalmak helyett az alkalmazás fogalmainak használata. A rendszer szóhasználata, kifejezései, jelölései, képi szimbólumai feleljenek meg a használóéinak. Az alkalmazás szokásainak követése azok természetes sorrendjében.

Meg kell ismernünk az adott szakterületet, mielőtt programot terveznénk számukra, hogy tisztában legyünk a kifejezésekkel, a szakmai zsargonnal, a rövidítésekkel. Ezekkel a felhasználók által ismert szavakkal címkézhetjük és jelölhetjük az ikonokat, funkciókat. Garantáltan nőni fog a felhasználói elégedettség, ha ismerős kifejezésekkel találkoznak a programban, és sikerélményük lesz a használattal kapcsolatban.

Bátorítsuk a kísérletezést:

Engedjük meg a műveletek visszavonását, nyújtsunk biztonságérzetet, véletlen hibák könnyen javíthatók legyenek. A szoftver egésze sohasem tanulható meg teljesen; a használó merjen - súlyos következmények nélkül - kísérletezni.

Ha ezeket a javításokat megengedjük, sokkal nagyobb kedvvel fogják a felhasználók a munkájukat végezni, hiszen tudják, hogy nem követhetnek el javíthatatlan hibát. Azokban az esetekben, amikor az adott művelet eredménye visszavonhatatlan, többször figyelmeztető üzenetet kell küldeni a használónak (felugró ablak, hanghatások), és megerősítést kérni a művelet végrehajtásához. Ha már kétszer is megerősíti, hogy szeretné az adott műveletet végrehajtani, már nem beszélhetünk véletlen hibáról.

Előzékenység

A kísérletezés ne lehessen végzetes. Motiválja a tanulást: egyszerűbb új ismeretek is nyújtsanak sikerélményt. (például új funkciók hiba nélküli végrehajtása segítség nélkül.)

A hibajelzések szövege sohase vádolja az embert. Tárgyilagosan állapítsa meg, hogy mi történt és tájékoztasson a megoldás lehetséges módjairól. Semmiképpen se az legyen a hibaüzenetben, hogy „Hibás formátumú fájlt próbált feltölteni, próbálja újra.”. Hanem e helyett: „A feltölteni kívánt fájl formátuma nem megfelelő. Kérem, ellenőrizze az állományt.” Az első verzió vádlón hangzik, a második tájékoztat, és megoldást kínál a probléma megoldására.

Értelmetlen eseményekre - érzéketlen gombok, lapozás a szöveg végén túl - legyen egyszerűen érzéketlen, ne jelezzen hibát. Ha a használó kéri, ilyenkor lehet egyszerű, szolid hangjelzést (halk sípolás) adni, vagy ha az egérrel a gomb fölé áll, akkor egy kis szövegdobozba a „funkció itt nem elérhető” üzenetet megosztani vele. Egy lehetséges megoldás, hogy az érzéketlen, nem használható gombokat kiszürkítjük, így egyértelművé téve, hogy használaton kívül vannak.

Értelmes műveletek azonnali végrehajtása visszajelzéssel és a visszalépés lehetőségével.

Könnyű kapcsolódás más programokhoz; hívás és export-import

Egy programnak megadott adatok gyakran más programokból származnak, illetve az eredményeket más programok alakítják tovább. Az interaktív munka során esetleg többször is kellhet más programokkal adatot cserélni. Érdemes a már közismert állományformátumokat alkalmazni. Ha ezektől eltérünk, akkor biztosítanunk kell a megfelelő interfészeket a kommunikációhoz más programokkal.

A bomba-biztos kezelőfelület elve

A kezelőfelület egyetlen vezérlő-eleme sem vezethet a „levegőbe”. A programozás kezdetén minden látható vezérlőelem legyen halványított, vagy vezethet egy „Ez még nem

működik” üzenetre. Ezeket fokozatosan cseréljük le a tényleges megoldásra. A „még nem működik” ablakokból biztosítuk a visszalépés lehetőségét.

4.A feladat előzetes megfogalmazása

4.1 használhatósági módszerek

A szoftver életciklusának fázisait sokféleképpen szokták meghatározni, az egyik lehetséges mód:

1. Célkitűzés,
2. Követelmények; elemzés és specifikáció,
3. Tervezés; magas szintű (elvont) tervezés, részlet-tervezés
4. Megvalósítás,
5. Tesztelés,
6. Üzembe helyezés,
7. Követés.

Mindezen lépések természetesen a szoftver felületének megtervezésében is fontos szerepet játszanak. A fázisok egymásra épülnek, előkészítik a következő fázisba lépés, valamint ha egy fázisban hibát észlelünk, akkor vissza kell lépnünk az előző fázisba, hogy az felmerült hibát ki tudjuk javítani.

A szoftver ergonómiai szempontú tervezését és vizsgálatát szolgálják a használhatósági módszerek (usability engineering).

A szoftver ergonómiai szempontú tervezését nem lehet elkülöníteni magától a szoftver tervezésétől, be kell építeni a folyamatokba. Viszont, ezen folyamatok tervezéséről külön kell beszélni, hiszen sokan megfeledkeznek a fontosságáról, nem ismerik az eszközeit.

Tervezési módszerek: egy szoftver-technológiai folyamat keretében, a célkitűzések megfogalmazása, a követelmények elemzése, a szoftver használhatóságának tervezése, és a részletes tervekben megfogalmazott szoftver használhatósági vizsgálata.

Vizsgálati módszerek: egy elkészült szoftver használhatóságának vizsgálata. Egyszerre a szoftvernek egy-egy kiválasztott részét vizsgáljuk, gyakran csak egy prototípust, esetleg csak papíron meglévő, részletes tervezetet vizsgálunk, előre meghatározott szempontok szerint. Egy szoftver egyfajta „minősítése”, vagy – a szoftver-technológiai folyamat részeként végrehajtva – a tervezés vagy megvalósítás hibáinak kiszűrése.

Ezen módszerek és eszközök használata a tervezési folyamat során extra költségeket jelentenek a fejlesztő cégeknek. Éppen ezért keveset árulnak el a nagy fejlesztő cégek a saját tervezési módszereikről, a kisebb cégek pedig vagy csak kis mértékben alkalmazzák ezen módszereket, vagy költségcsökkentés miatt egyáltalán nem.

A vizsgálatok és a tervezés módszerei többnyire egyszerű, kézenfekvő gondolatokon alapuló eljárások, amelyeket a velük szerzett tapasztalatok (mire, hogyan használhatók), és a módszeres végrehajtásukból adódó eredményességük avat „módszerekké”. Ahhoz hogy a jövőben még sikeresebben alkalmazzhassuk a módszereket, írjuk le tapasztalatainkat (fontos az írásban rögzítés), így eredményesebbek lehetünk a jövőben. Amikor első alkalommal használjuk a módszereket, ajánlatos mások tapasztalataira támaszkodnunk, és a sajátjainkat feljegyezni.

Minden módszer alkalmazása három, egymás utáni fázist tartalmaz: az előkészítés, a végrehajtás és az értékelés fázisát.

Előkészítés:

Válasszuk ki, hogy melyik módszert fogjuk alkalmazni, és ismerkedjünk meg vele.

Készítsünk vizsgálati tervet: rögzítsük a vizsgálat célját, a szükséges eszközöket, a folyamatokat, miképpen fogjuk rögzíteni az eredményeket, hogyan fogunk értékelni.

Előkészületek megtétele: a projektben résztvevő személyek kiválasztása és felkészítése, időpontok helyszínek egyeztetése, értékelési elvek meghatározása, feladatlapok és jegyzőkönyv űrlapok elkészítése.

Végrehajtás

A vizsgálat végrehajtása a vizsgálati terv szerint, vizsgálati jegyzőkönyv készítése (szigorúan írásban), felvételek készítése. A vizsgálat lefolytatása során események és eredmények tárgyszerű rögzítése időpontok megjelölésével.

Értékelés:

A feljegyzések rendszerezése, jegyzőkönyv, értékelés, következtetések, írásos jelentés, dokumentumok irattárban elhelyezése.

Fontos hangsúlyozni, hogy mindent írásban kell rögzíteni, mert néhány év múlva nem biztos, hogy emlékeznénk, miért döntöttünk anno úgy, ahogy. A vizsgálat után a tapasztalatokat rögzítsük későbbi használatra.

Felmerülhet az olvasóban, hogy mindez szépen és jól hangzik, de rendkívül sok időt és energiát emészt fel a rengeteg dokumentáció készítése, írásba foglalása. Viszont a következő projekt alkalmával nagyon meg fogja könnyíteni a munkát, hogy nem kell mindent a nulláról kezdenünk, és rendelkezésre áll jó néhány összeszedett, hasznos dokumentáció.

Amit eddig még csak „módszerként” említettünk, arra nézzünk néhány példát:

- Kérdőíves módszer
- felmérés
- interjú
- fókusz csoport
- adott szempontok szerinti értékelés
- bejárás módszerek, megfigyelési módszerek
- kommentált munka.

Ezek a fogalmak nem ismeretlenek azok számára, akik bármilyen tervezési folyamatban részt vettek már. A kérdőívezés, a felmérés és az interjú a legalapvetőbb eszközei a tervezési módszereknek, hiszen kérdezés nélkül nehezen jutnánk előre. Az itt kapott eredményeket több szempont szerint értékelhetjük. A megfigyelés elég időigényes módszer, viszont nagyon hatékony a kommentált munkával egyetemben. A módszereket egy későbbi fejezetben részletesen tárgyaljuk.

A szoftver tervezésén és előállításán belül is meg kell különböztetni a kezelőfelület (user interface), illetve a program belső adattárolásának és működésének tervezését. Ez a kettő más-más ismereteket és szemléletet igényel.

Viszont sosem szabad megfeledkeznünk arról, hogy a tervezés emberközpontú maradjon.

Az ember központú tervezés három alapelve: kiindulás és állandó fókusz: a használó személye és feladatai, a szoftver-használat kísérleti vizsgálata, iteratív tervezés; a tervezés, megvalósítás, tesztelés, és javítás ismételt alkalmazásával.

Az emberközpontú tervezés "eredményeként... az emberek a kívánt műveleteket, szolgáltatásokat, és támogató feladatokat minimális stressz mellett maximális hatékonysággal végezhetik el".

4.2 Feladat előzetes megfogalmazása, feladatot meghatározó mondat

A feladat előzetes megfogalmazásának célja a feladat jobb megértése, a projekt körvonalainak rögzítése, védekezés a megrendelő utólagos ötletei és saját eltéréseink ellen. Szoftvert akkor készítünk, ha igény merül fel valamilyen probléma megoldására. Célkitűzéseinket írásban rögzítjük. Ez az előzetes megfogalmazás a későbbi folyamatok során sokat változhat, javulhat, gyarapodhat, és nem utolsósorban részletesebbé válik.

A feladat előzetes megfogalmazása rövid, néhány oldalas irat, amelynek célja az igények első, előzetes megfogalmazása. Alapvetően a szoftver használhatósági tervezésének eszköze. Alkalmazható kész szoftver termékekre is, és hamar fény derülhet, arra, hogy a már kész szoftver tervezésekor milyen hibákat vétettek.

Egy szoftver tervezésének legelső lépése a célkitűzés rögzítése, az ötlet megfogalmazása, a vízió. A célkitűzések írásba foglalásának sok előnye van. Sem a megrendelő, sem a fejlesztő nem feledkezik meg az eredeti célokról, az egyik nem követel többet, a másik nem nyújt kevesebbet a rögzítetteknél. Természetesen ezen célkitűzések részletes megfogalmazása csak később történik. A feladat előzetes megfogalmazása attól lesz jó és hatékony, hogy megköveteljük magunktól az ott leírtak betartását.

A feladat előzetes megfogalmazása általános esetben három részből áll:

1. A "helyzet" leírása; röviden, 1/2 - 1 oldalon, a körülményekről, amelyek a változtatást szükségessé teszik.

2. A feladatot meghatározó mondat, amelynek négy kötelező része: E (ember), T (tevékenység), F (forma), H (használhatósági célkitűzések); tömören, tagoltan és tárgyszerűen (ez készül el utoljára!)

3. A feladatot meghatározó mondat (négy részének) kifejtése néhány oldalon

A helyzet leírása:

Megfogalmazzuk a problémát néhány mondatban és az esetleges megoldásra is javaslatot teszünk.

Például: Az autósiskolákban a tesztelés gyakorlása során az okosabb tanulók unatkoznak, mert a papír alapú tesztekkel gyorsan végeznek.

Bővebben kifejtve:

Az oktató kiosztja a rendelkezésre álló néhány különböző papír alapú tesztet a tanulóknak, de az okos tanulók hamar végeznek és az óra hátralévő részében unatkoznak.

A feladatot meghatározó mondat:

A feladatot meghatározó mondat a szoftver iránti igény négy összetevőjét fogalmazza meg egyetlen tömör mondatban:

- (E) az embert, a szoftver használóját,
- (T) a használó által végzett tevékenységet, amelyhez a tervezett szoftver segítséget kell, hogy nyújtson,
- (F) a megoldás módját, formáját, és
- (H) a szoftverrel kapcsolatos használhatósági célkitűzéseket.

Ez a mondat sűríti össze a project egészét, megfogalmazásához rengeteg információra és előzetes anyaggyűjtésre van szükség.

A fent említett példa-probléma feladat meghatározó mondata így hangozna:

Oktatóprogram készítése (F), amelynek segítségével az autósiskola tanulói (E) maguktól is rövid idő alatt (H) képességeikhez mérten elegendően sok, változatos tesztet tudnak megoldani, és a feladatok megoldásában kellő gyakorlatra tehetnek szert (T).

Még néhány példa probléma és feladat meghatározó mondatra:

Probléma:

Az egyetemen elhelyezett ital-automata csak fémpénzt fogad el, így a csak bankjeggyel rendelkező hallgatók és oktatók nem tudják használni az automatát.

Feladat meghatározó mondat:

Olyan pénzbedobós automata (F) tervezése ital vásárlására (T), amellyel a hallgatók és oktatók (E) könnyen és gyorsan italt tudnak vásárolni papírpénz használatával is. (H).

Probléma:

A Bank ügyfelei a bank honlapján nem tudnak tájékozódni a svájci frank napi középárfolyamáról, és így nem tudják kikalkulálni következő törlesztő részletük összegét.

Feladat meghatározó mondat:

Olyan menüpont létrehozása a bank webalkalmazásában (F), amelynek segítségével egy ügyfél (E) könnyen és gyorsan, dátum kiválasztásával (H) megtudhatja az adott napi svájci frank középárfolyamát (T).

A feladatot meghatározó mondat kifejtése:

Mire az előbb megfogalmazott példa mondatokat leírhatjuk, rengeteg előkészülettel nézünk szembe. A mondat megfogalmazását valójában a négy részére (F, E, H, T) vonatkozó részletek összegyűjtése előzi meg. Ezen négy összetevő kifejtésével teljesebb képet kapunk a megfogalmazandó feladról. A mondat tömörsége miatt általános megfogalmazást ad csupán, de nem is tiszte a részletes leírás. A részletes leírást a következőkben tárgyaljuk.

4.3 A feladatot meghatározó mondat kifejtése

A feladatot meghatározó mondat kifejtésének négy része:

1. (E) Ember, a használó

Az ember áll a középpontban, ő a cél és a kiindulópont is. A mondatban az embert mindig egy szóval illetjük (tanuló, utas, felhasználó).

Jelen esetben az ember számunkra fontos tulajdonságait a következő csoportokba sorolhatjuk:

- számítástechnikai ismeretei és gyakorlottsága,
- szakmai ismeretei és gyakorlottsága,
- szerepe a munkahelyi szervezetben,
- releváns emberi tulajdonságai: kreativitása, önállósága, nyitottsága,
- stb.; ide tartoznak: a gyengén látás, és más korlátozó tényezők is.

Egy programnak többféle felhasználója is lehet. Nekünk minden lehetséges felhasználó típust számításba kell vennünk.

A fent említett tulajdonság csoportok szerint lehetséges, hogy a programunkat szintén programozók vagy erős informatikai tudással rendelkező emberek fogják használni, de fel kell készülnünk kezdő szintű felhasználókra is (akik cserében az informatikai hiányosságokért szakértők azon a területen, ahová a terméket készítjük).

Miután felmértük a tulajdonságokat személyesen is meg kell ismerünk a felhasználókat a már említett eszközökkel (interjú, megfigyelés, stb.).

A fejezet eleji példákra hivatkozva, feltehetjük például, hogy egy autósiskola tanulói már használtak számítógépet, alapismereteik vannak az operációs rendszerekkel, de legalább is ismerik a periféria eszközöket. Egy ilyen programot a lehető legegyszerűbben használhatóra terveznek éppen azért, mert idősebb, „nem a számítógépeken nevelkedett” generáció képviselői is szeretnék jogosítványt, és gyakorolni a vizsgára a szoftver segítségével.

A másik példában említett italautomata esetében is feltételezzük, hogy mindenki ismeri a pénzt, viszont ügyelnünk kell arra, hogy a gombok az alacsony emberek számára is elérhető magasságban legyenek, sőt ha egészen jó automatát szeretnénk tervezni, akkor tekintettel kell lennünk a kerekesebb emberekre is, és egészen alacsonyra elhelyezni a nyomógombokat és a kiadóablakot.

Szoftvertervezésnél tehát tekintettel kell lennünk a felhasználók különbözőségére és alternatív megoldásokat nyújtani mindenki számára a jobb használhatóság és a nagyobb felhasználói elégedettség érdekében.

2. (T) Tevékenység

Szoftver ergonomiai szempontból a „mit csinál a felhasználó?” kérdésre speciális a válasz. A felhasználó tevékenységét írjuk le. Ezt a leírást semmiképpen se a számítástechnika szakszavaival tesszük meg, hanem a felhasználó által is értett szavakkal. Használjuk az ő területének fogalmait (ezekkel a fogalmakkal nekünk is meg kell ismerkednünk, ahhoz, hogy jó szoftvert tervezhessünk). Ha egy meglévő szoftver felületét vizsgáljuk, akkor láthatjuk, hogy melyek azok a fogalmak, amelyeket már megszoktak a felhasználók (még ha az ő szakterületükön kívül is esnek), ezeket nem célszerű megváltoztatni, mert könnyen összezavarhatjuk a felhasználókat és az új felület tanulása sokkal nehezebben fog menni.

3. (F) Forma

A megoldás formája sokban függ a megrendelő lehetőségeitől (anyagi, tárgyi lehetőségek). Például, ha nem tudnak beruházni új számítógépekbe, vagy megvásárolni egy operációs rendszert, akkor nekünk kell igazodni a meglévő eszközökhöz, és a lehetőségekhez mérten a leghatékosabban tervezni a szoftverünket. Mindezek, és a cég munkatársai ismeretét felmérve kezdhet körvonalazódni bennünk, hogy milyen platformra, milyen fejlesztői eszközökkel készítsük el a szoftvert.

4. (H) Használhatósági célkitűzések

Még mindig nem az a kérdés, hogy mire akarják az elkészült szoftvert használni, hanem, hogy mennyire jól használható a termék.

4.4 Célkitűzések

Mint ahogy már említettem a használhatóság szempontjai négy csoportba oszthatóak:

- hatékonyság
- biztonság
- funkcionális teljesség
- emberi minőség

A célkitűzések szolgálják azt, hogy az eredmény mindig jobb legyen, mint ahonnan elindultunk. Ha egy meglévő felületet tervezünk újra, akkor a fenti szempontokat vizsgálva (pl. mérések) jobb eredményeket kapjunk.

A célkitűzéseket úgy kell megfogalmazni, hogy azok egy szoftverben tárgyilagosan vizsgálhatók legyenek. Ezért törekedni kell „mérhető” szempontok megfogalmazására; a "mérés" eredménye néhol fizikai mérésekből származik, másutt a használók véleményéből vonható le. Vannak olyan szempontok is, amelyek nem mérhetők.

Szempontok célkitűzések megfogalmazásához:

- kezdeti teljesítmény, tartós teljesítmény
- megtanulhatóság, megtarthatóság (az ismeretek megtarthatósága)
- összetettebb funkciók használata
- első benyomás, hosszú távú megelégedettség
- hibalehetőségek, a hibák gyakorisága, a hibajavítás módja, a javítás ideje, egyszerűsége
- testre-szabás, egyéni alakíthatóság, a munka átszervezhetősége
- szubjektív elégedettség
- feszültség (frusztráció)
- munkatartalom (sokoldalúság)
- az ember önértékelése

- a rendszer iránti bizalom

Mérés módjai lehetnek:

- egy-egy feladat teljesítésének ideje
- adott idő alatt teljesített feladatok száma
- a hibák száma adott idő alatt, gyakorisága
- siker/hiba arány
- a hibák javítására fordított idő
- parancsok, műveletek száma egy feladat végrehajtásában
- a súgó/dokumentáció használatának gyakorisága
- a nem használt parancsok száma
- az elégedettség vagy elégedetlenség kifejezésének gyakorisága

Számszerűen:

- gyorsabb tanulhatóság (régen két hét, most 1 hét)
- funkciók gyorsabb végrehajtása (eddig egy munkafolyamat 20 másodpercig tartott, most) 5 másodperc alatt kapjuk az eredményt.

4.5 A követelmények elemzése

A feladat előzetes megfogalmazásában rögzítettük elképzeléseinket, terveinket, célkitűzéseinket. A követelmények elemzésének befejezése után módosítjuk a feladat előzetes megfogalmazásában rögzítetteket az eredmények függvényében.

- A tevékenységek, folyamatok elemzése alapján megalkotjuk azok modelljét, azonosítjuk azok elemeit és ezek egymásra épülését, a tevékenységek hierarchiáját. A modellben is ügyelünk arra, hogy a felhasználók által ismert, az adott szakterület szavait használjuk, hogy mindenki megértse, mit is szeretnénk megvalósítani.
- Megismerjük a használókat, ismereteiket és készségeiket, az általuk végzett tevékenységekkel kapcsolatban. Felmértük a szaktudásukat, számítástechnikai ismeretüket, ezekhez igazodva tesszük meg a következő lépéseket.
- Körvonalazzuk, majd pontosítjuk a megoldás formáját. Esetünkben ez mindig valamilyen számítástechnikai rendszer, amelynek típusa és kiépítése lehet helyi adottság, vagy megbecsülhető a tevékenységek igényeiből. Valamint felmértük a megrendelő anyagi lehetőségeit, számítógépeinek állapotát, a rendelkezésre álló operációs rendszert.
- A használhatóság – hatékonyság, biztonság, funkcionális teljesség, és emberi minőség – szintjének megfogalmazása tartalmazhat elemi követelményeket (pl. ne rontsa az ember szemét, adott típusú vészhelyzetben 2 másodpercen belül megoldás), részben a korábbi rendszer paramétereit meghaladó követelményeket, amelyek pénzügyileg alátámaszthatják az új megoldást. Ezeket a javításokat, újításokat számokkal is alátámasztjuk, így könnyebben meggyőzhetjük a megrendelőt igazunkról és a szoftverünk jóságáról.

4.6 Használhatósági célok a feladatok előzetes megfogalmazásakor

A használhatóság számunkra továbbra is azt jelenti, hogy mennyire jól használható egy szoftver, nem pedig hogy mire használható.

A használhatóság összetevői:

- Hatékonyság, eredményesség
- Megbízhatóság, biztonság
- funkcionális teljesség
- Emberi minőség

A hatékonyság és eredményesség arra vonatkozik, hogy a felhasználók, milyen hatékonyan tudják használni a szoftvert.(milyen gyorsan tanulják meg kezelni, milyen gyorsan tudják rajta végrehajtani az egyes feladatokat.) Sok esetben gazdasági hatékonyságról is beszélhetünk, hiszen, minél gyorsabb a munkavégzés (a szoftver könnyű használhatósága miatt), annál nagyobb lehet bevétele cégnek.

Kritikus folyamatirányítási rendszereknél megbízhatóság helyett biztonságról beszélünk.

Funkcionális teljességről beszélünk, ha a felületen nem futhatunk nem használható menüpontokba, ikonokba.

Az emberi minőség összefoglaló elnevezése olyan szempontokat takar, amelyek elsősorban az embert (a használót) érintik, de rajta keresztül gazdasági hatásai is vannak. Ide sorolhatók:

- egyhangúság, illetve annak hiánya
- megismerhetőség, megtanulhatóság, megtartás és fölidézés
- frusztráció
- kezdeményezés
- fejlődés
- szellemi és fizikai terhelés, fáradás
- munka-tartalom
- hibatűrés
- hasznosság

- könnyű használat
- attitűd (az ember érzései, véleménye a szoftverről)

4.6.1 Használhatósági terv

A fent leírtakat összegeznünk kell egy ún. használhatósági tervben, amelyben szerepelnek a vizsgálandó használhatósági szempontok, a mérési feladatok, a meglévő rendszerben jelenlegi érték, a cél-, az elfogadható-, és a legjobb érték. Majd az új rendszerben elvégzett mérési eredményekkel bővíthetjük a használhatósági tervet, ezzel is szemléltetve, hogy az értékek javultak vagy sem.

Összességében tehát azt mondhatjuk, hogy a feladat előzetes megfogalmazása rögzíti előzetes ismereteinket, elképzeléseinket.

A vizsgálatok során dokumentumokat készítünk, melyekben tömören, tárgyyszerűen fogalmazunk a későbbi jól használhatóság érdekében.

A vizsgálatok után megírjuk a feladat meghatározó mondatot, amely összesűrítve tartalmazza a javítani kívánt problémát.

A feladat előzetes megfogalmazása egy már meglévő rendszer esetében, annak átalakítási folyamatát írja le (emberi tényező szempontjából), egy új rendszer esetében hatástanulmány készítése mindenképpen ajánlott.

5. Felhasználók megismerése, módszerei

Az eddigi fejezetek során rögzítettük a célkitűzéseinket, elkészítettük a feladat előzetes megfogalmazását. Elemeztük a követelményeket, és megállapítottuk, hogy az elemzés után módosítanunk kell a feladat előzetes megfogalmazásán a kapott eredmények függvényében. Vajon a követelmények elemzése hogyan néz ki az ember, a használó szemszögéből? Miért fontos ezt vizsgálnunk? Hogyan ismerhetjük meg a felhasználókat? Ezen kérdésekre keressük a válaszokat.

A követelmények elemzése a használó szempontjából az ő tevékenységének, feladatainak elemzését jelenti.

A felhasználók egyes csoportjainak elemzési eredményét felhasználói profilnak nevezzük. A felhasználókat több szempont szerint csoportosíthatjuk, és a valamilyen tulajdonság szerint egy csoportba soroltakhoz készítünk egy profilt.

A módszerek, melyekkel a felhasználókat megismerhetjük, legtöbb esetben irodai környezetre specializálódtak, de kis módosítással adaptálhatók szoftver környezetbe is.

5.1 Vizsgálati módszerek

A leggyakoribb vizsgálati módszerek:

- kérdőíves vizsgálat
- interjú
- személyes beszélgetés
- célzott, csoportos megbeszélés (fókusz csoport)
- megfigyelés
- kommentált műveletek
- szerep-átvétel

Ezek a módszerek egyszerű vizsgálatokon alapulnak, és csak sokszori alkalmazás után avathatjuk őket módszerré. Első alkalommal mások tapasztalataira kell támaszkodnunk, míg sajátot nem szerzünk.

Eddig a vizsgálatról nem ejtettünk szót részletesen, lássuk miből is épül fel valójában egy vizsgálati terv szerint lefolytatott vizsgálat.

Az első és legfontosabb kérdésünk, hogy mi is vizsgálunk? Természetesen vizsgálatunk középpontjában mindig az ember áll, tehát az ő tevékenységüket, ismereteiket, szokásaikat vizsgáljuk.

A felhasználók egy csoportját vizsgálva, az egyes tulajdonságokat nézve, az egyes emberek eltéréseket mutatnak. Egy csoport tagjait először próbáljunk egy szóval jellemezni, Ez a jelző legtöbb esetben a munkakörükkel egyezik meg. Például egy csoporton belül a rendszeres számítógép használók aránya 50%. Viszont tudjuk, hogy a csoport 80%-a könyvelő. Nem várjuk tehát el az összes könyvelőtől, hogy magas szintű számítógép használói ismeretekkel rendelkezzen. Ezek nagyon fontos információk lesznek a későbbiekben, hiszen minél kevésbé gyakorlottak a felhasználók a számítógéppel, annál inkább törekednünk kell a könnyű használhatóságra, és arra, hogy a felületünk hasonlóságot mutasson a mindennapi rutinjuk alatt használatos eszközeikkel, űrlapokkal, kifejezésekkel.

A használóra vonatkozó ismereteink:

- Személyes előzmények: iskolai végzettség: legmagasabb iskolai végzettség, szakirány, tanult más tárgyak. Munkahelyek: a jelen és korábbi beosztások, felelősségek, oktatások, eltöltött idő ezen a helyen.

Ha a felhasználó iskolai végzettsége alapján nem is feltételeznénk a számítástechnikai ismereteket, ennek ellenére szerezhettek oklevelet tanfolyamon, vagy be nem fejezett iskolai tanulmányokat folytathatott, ahol szintén lehettek olyan tantárgyak, amelyek elősegítették ezen készségek kifejlődését. Nem is beszélve az előző munkahelyeken szerzett tapasztalatokról, amelyek esetleg már elhalványodtak a használó emlékezetében, de tudva azt, hogy egy kis ismétléssel ezen tudás felszínre hozható, hamarabb meg tudjuk neki tanítani a szoftverünk használatát.

- Számítástechnikai ismeretek, gyakorlottság és szokások; (kezdő, használó, programozó, rendszergazda, illetve: kezdő, eseti használó, egy alkalmazásban gyakorlott, többen is, mióta, milyen gyakran használ számítógépet?).

A kezdőkkel és az eseti felhasználókkal lehet a legtöbb problémánk gyakorlatlanságuk miatt, de ezen tudás birtokában szervezhetjük számukra a tréningeket. A gyakorlott felhasználók esetében meg kell vizsgálni, hogy milyen típusú rendszeren végezték eddig a munkájukat. Ha hasonló területre fejlesztjük jelenlegi szoftverünket, akkor esetleg információt szerezhethünk ezen felhasználtól arra vonatkozóan, hogy mi volt az, amit szeretett a régen használt szoftverben és mit nem, és a szeretett dolgokat beépíthetjük a mi szoftverünkbe is.

- Számítógépes tapasztalat: a tapasztalat ideje, a géphasználat gyakorisága, a használt számítógép- és periféria típusok, operációs rendszerek, a használt interakciós stílusok (GUI, parancsnyelv. Táblázatkitöltés, stb.)

Természetesen nem mindegy, hogy a használók a múltban naponta mennyi időt töltöttek számítógépes munkával. Ha csak néhány hónapos tapasztalatról van, szó, heti egy-két alkalmas géphasználattal, akkor számítsunk arra, hogy ezek a felhasználók sem elég gyakorlottak, és betanításuk több időt fog igénybe venni. Manapság egy átlagos felhasználó számára a legkényelmesebb felület az ún. „kattingatós” felület, ahol csak néhány mezőt kell kitölteni. Ez jól működhet egy könyvelőirodánál, vagy például az adóhivatalnál.

- Szoftverhasználattal kapcsolatos tapasztalat: a tapasztalat ideje, a használat gyakorisága, feladattípusok, milyen gyakorisággal, a használt szoftverek; ezen a munkahelyen és másutt.

Nem csak a munkahelyi szoftverek használata során szerezhethetnek a felhasználók tapasztalatokat. Megkérdezhetjük, hogy otthon milyen programokat használnak, melyiket szeretik, melyiket nem, melyik funkciót látnák szívesen a munkahelyi szoftverben is. Ez elég jó módszer arra, hogy mások (más programozók) tapasztalatát felhasználjuk a saját szoftverünk fejlesztésekor, hiszen az általuk jól kidolgozott elemeket, a felhasználóink tapasztalata alapján beépíthetjük a saját szoftverünkbe.

- A szakmával és feladataival kapcsolatos iskolázottság, ismeretei, gyakorlottsága, fogalmai, szóhasználata, szokásai, helye a munkamenetben, motivációja.

Mint már említettem, fontos az, hogy a számítástechnikai szakszavak helyett a felhasználóink munkaterületének szakszavait használjuk a szoftverünk felületének készítéskor. Érdeklődhetünk, hogy mi nyújtana nagyobb felhasználói élményt (még ha egy munkahelyi szoftverről is van szó), mi lenne az a tényező, ami miatt nagyobb kedvvel látnának munkához.

- Általános motiváció, önállóság, manuális készségek (a személyiségi jogok és az emberi személyiség tiszteletben tartásával). Személyes előzmények: életkor, nem, a számítógépekkel (a termékkel) kapcsolatos attitűd, jobb-balkezes, tanulási preferenciák, az új technológiákkal kapcsolatos attitűd

Ezekkel a témákkal óvatosan kell bánnunk, nehogy megsértsük a felhasználók személyes szféráját. Vannak, akik szívesen beszélnek készségeikről, képességeikről, tanulási problémáikról, de vannak, akiket zavar, ha hibáikra fény derül. Ha látjuk, hogy a kérdések kellemetlenül érintik a felhasználót, próbáljuk más irányba terelni a kérdéseink vonalát.

- Feladatai, tevékenysége: célok, feladatok, műveletek és ezek szerveződése, összefüggések, kapcsolat az emberi és szervezeti környezettel, a feladatok lényeges vonásai, hibaforrások.

Érdeklődjunk jelenlegi feladatai iránt. A munkája milyen közvetlen hatással van a munkatársai feladataira? Milyen hatással lehet munkatársai feladatai sikerességére? Milyen gyenge pontokat vél felfedezni a jelenlegi munkájában? Mik a lehetséges hibázási lehetőségek?

- Milyen oktatás, betanítás látszik szükségesnek.

Ha a fenti összes kérdésre megkaptuk a válaszokat, akkor körvonalazódik bennünk, hogy az egyes felhasználói csoportoknak mennyi időre lesz szüksége az új szoftver megtanulásához. Látható lesz az is, hogy milyen szintű tréningekre lesz szükség. A felhasználókat csoportokba oszthatjuk aszerint, hogy kinek kell átfogó tréning, kinek elegendő csupán egy egyszerű szoftver bemutatás.

5.1.1 A kérdőíves módszer

A kérdőív lehet papír alapú vagy elektronikus. Az elektronikus kérdőív feldolgozása könnyebb, de vannak helyzetek, amelyek megkívánják a papíros formát. A célunk ismeretek szerzése a leendő felhasználókról, tevékenységükről, munkájukról.

A kérdőívek általában személytelenek, olyan emberekkel kell kitölteni, akik közlékenyek, nyitottak, motiváltak. Azok az emberek, akik nem rendelkeznek ezen tulajdonságokkal, talán nem valós válaszokat adnának a kérdésekre, nem kell ők odafigyeléssel töltenék ki a kérdőívet, ezzel akaratlanul is félrevezetve bennünket.

A kérdőív összeállításakor figyelni kell a következőkre:

- Előre meggondolni: mire vagyunk kíváncsiak, hogyan fogjuk kiértékelni, hogyan fogjuk fölhasználni.

Itt lehet gondolni, arra, hogy papír alapú vagy elektronikus legyen-e a kérdőív, milyen témakört szeretnénk körüljárni, hány kérdés az, amire még szívesen válaszolnak.

- Kiket akarunk megkérdezni?

A felhasználókat csoportokba osztottuk végzettségük, tapasztalatuk alapján. Egy alacsonyabb (számítógépes) végzettséggel rendelkező felhasználónak más kérdéseket teszünk fel, mint például egy rendszergazdának, vagy más kérdést teszünk fel az eseti felhasználóknak, mint a napi rendszerességgel a szoftverrel dolgozóknak.

- Általában név nélkül („a név beírása nem kötelező”), esetleg megkülönböztető jelzéssel.

Fontos, hogy a kérdőív kitöltője megőrizhesse anonimitását, ugyanis név nélkül sokkal közlékenyebbek az emberek az élet más területein is, az anonimitás mögé bújva sokkal őszintébben megmondják, ha valami nem tetszik neki, vagy valamivel nem értenek egyet.

- A kérdőív feleljen meg néhány formai és tartalmi szabálynak.

Mint minden hivatalos irat, dokumentum, a kérdőívnek is meghatározott formát kell követnie és a tartalmának is szabályozottnak kell lennie.

- A megfogalmazott kérdőív ellenőrzése szempontok listája szerint.

A megfogalmazott (még kitöltetlen) kérdőívet ellenőriznünk kell, mielőtt a kitöltőkhöz kerülne. A szempontok lehetnek: körültekintően tettük-e fel a kérdéseket, biztosan kapunk-e átfogó képet a feltett kérdések alapján, egyik kitöltőt se fogja sérteni egyik kérdés sem stb.

- Tervezési döntéseink írásban rögzíthetők, indokolhatók.

- A kérdőívet először magunk, majd kisebb csoport megpróbálja kitölteni.

Próbakitöltést végzünk, tesztelve azt, hogy a kérdőív megfelelő, a kérdések érthetőek, világosan vannak megfogalmazva. Esetleges hibák felfedezése miatt, a kérdőívet javíthatjuk és újratesztelhetjük.

- A hely és időpont kiválasztása: ráérés és összpontosítás. Egyéni, otthoni kitöltés, vagy együtt. Internet? Az emberek munkahelye előnyös.

A hely és időpont választása függ a kérdőív hosszától. A nem túl hosszú kérdőíveket célszerű a munkahelyen kitölteni, a hosszúakat otthoni kitöltésre átadni. Az otthoni kitöltés azt a veszélyt rejtheti, hogy a kitöltő esetleg dekoncentrált (megzavarják, egyszerre több dologra kell otthon figyelnie stb.), valaki segít neki a kitöltésben, és ez által nem kapunk a kiértékelés után objektív képet.

A munkahelyen történő kitöltés is kétféle lehet: egyéni vagy csoportos. Az egyéni kitöltés előnye, hogy szubjektív véleményeket kapunk, míg a csoportos kitöltés előnye, hogy egy közösség egybehangzó véleményét kapjuk eredményül.

A kérdőív kitöltésére kiválasztott embereknek megfelelő időpontot kell biztosítanunk, amikor megfelelően tudnak koncentrálni. A javasolt időpont a hét közepe, lehetőleg a délelőtti órákban, amikor még nem fáradtak az emberek túlságosan, és kedvvel, odafigyeléssel töltik ki a kérdőívet.

Miután összeállítottuk a kérdőívünket, javasolt egy ellenőrzés végzése az alábbi szempontok szerint:

A kérdések szempontjából:

- egyszerűek, érthetőek a kérdések?
- egyértelmű a válaszadás módja?
- könnyen kitölthető?
- nincs felesleges kérdés?
- könnyen értékelhető?

A kérdőív szempontjából:

- könnyen kitölthető?
- nem túl sok a kérdés?
- terjedelme nem túl nagy?
- az elején kijelenti a kérdőív célját?
- van felvilágosítás a kitöltés módjáról?
- logikusan csoportosítottak a kérdések?

A kérdések fajtái a válaszadás szerint lehetnek:

- igen-nem kérdés
- felsorolások – aláhúzással vagy bekarikázással, sorszámozott alternatívákkal, (alig, közepesen, nagyon)
- kizáró- és többszörös választás, kiegészítő szavak a felsoroltakon kívül
- számszerű fokozatok: 0 vagy 1-től 5-ig (megmondva a 0 és az 5 jelentését, a fokozatok természetes értékére, sorrendjére ügyelve)
- ellentétpárok
- szám adatok, egy-egy szó
- szöveges válaszokat nehéz értékelni, ezért célszerű a fenti kérdésfajtákat választani.
- „A kérdés nem megválaszolható” jelzés (ha kell). Például, ha egyik kérdés következik a másiktól, és az elsőre nemleges választ adott a kitöltő, akkor lehet nincsen értelme a következő kérdésre válaszolnia.

Ha meggyőződünk arról, hogy a kérdőívünk megfelel minden szempontból, akkor kitöltethetjük a célszemélyekkel a fent említett csoportos vagy egyéni módon, illetve megfelelő helyszín és időpont kiválasztása után.

5.1.2 Interjú

Az interjút, annak érdekében, hogy nagyobb sikert érjünk el vele, megelőzheti egy kérdőív kitöltése, így felkészülhetünk a használó-specifikus kérdésekkel.

Az interjú lényege, hogy személyesen találkozzunk a felhasználóval. Gyors és egyszerű módszer, azonnali eredményekkel, sok és mélyreható ismereteket szerezhethetünk. Arról tájékozódunk, hogy most mit csinál, nem arról, hogy mit vár az új szoftvertől. Egy jól sikerült interjú meghatározhatja a későbbi együttműködés jó légkörét, de természetesen egy rosszul sikerült interjú a fordítottját eredményezheti.

Előzetesen egyeztessünk időpontot és helyszínt.

Fontos, hogy az interjú során, amennyiben a másik fél is beleegyezik, használjunk hangfelvevő készüléket. Ugyanis a sűrű jegyzetelés megakaszthatja az interjút, szaggatottá teheti, és a használó „kihallgatáson” érezheti magát. Így utólag elemezhetjük a hallottakat. Biztosítanunk kell azonban a másik felet, hogy a felvételt törölni fogjuk, valamint azt is, hogy ő is meghallgathatja a felvételt, ha szükségesnek érzi. Fontos a bizalom megtartása a későbbi biztosabb együttműködés érdekében.

Az interjú időtartama ne legyen hosszabb, mint egy óra. Ennyi ideig lehet koncentrálni teljes körűen a kérdésekre, de hosszabb idő után már az interjú alany fárad, várja, hogy vége legyen, nem ad pontos válaszokat (akár önhibáján kívül is).

Az interjú lehet előre tervezett szerkezetű (kérdések és válaszok), vagy szabadon folyó, de mederben tartott beszélgetés. Az interjú elején, esetleg az első interjúban, kiderül, ha a személy túlzott zárkózottsága, vagy túlzott közlékenysége miatt fokozottan kell előmozdítani, vagy korlátozni.

Ha az interjú során újabb kérdések merülnek fel bennünk, akkor kérjük meg a használót, hogy néhány nap múlva szánjon ránk újra egy kis időt, és tegyünk fel neki az újabb kérdéseinket

egy másik alkalommal. Addigra mi is átgondolhatjuk, hogy mit is szeretnénk még pontosan megtudni.

Az interjú vázlatos felépítése:

- az interjú célja, megismerkedés, gátlások oldása
- a feladatok, a legfőbb tevékenységek felsorolása
- ezek részletezése egyenként
- munkamódszerek
- kapcsolódások
- teljesítmények

Az interjú során tehát célunk:

- a használó megismerése
- képet kapni a munkájára vonatkozó ismereteiről, fogalmairól
- megismerni a szóhasználatát, mentális modelljét
- napi rutin feladatainak felsorolása, ezek tartalma
- munka módszerének megismerése
- teljesítmény adatok
- a feladatok végrehajtási módjának megismerése
- műveletelemek.

5.1.3 Személyes beszélgetés

Az interjúnál kötetlenebb forma, legtöbbször spontán kialakuló beszélgetés (ebéd, kávézás közben). Amikor mindkét fél éppen ráér, akkor általános kérdéseket tehetünk fel, ne legyenek a kérdéseink között mélyrehatóbbak. Jegyzetelni nem illik ilyen helyzetben, de a fontos dolgokat jegyezzük meg és később írjuk le.

Fókuszcsoporthoz megbeszélés:

A használók, a tervezők és a fejlesztők közös megbeszélése határozott időközönként.

Célja, hogy képet kapjunk az aktuális helyzetről.

Előnyei: több személytől több szempont várható, minden érdekelt kifejtheti a véleményét, jó közérzetet teremt.

Hátrányai: több időt vesz igénybe, a domináns személyiségek elnyomják a csendesebbeket.

5.1.4 Megfigyelési módszerek

- Passzív jelenlét a valódi munkahelyzetben. A megfigyelő nem zavarhatja a munkát, a háttérben ül. Ez azt jelenti, hogy amíg a használó dolgozik figyeljük a munkáját, szigorúan csendben. Így láthatjuk, hogy hová kattint, miket használ, melyek azok a funkciók, amelyek segítik az ő munkáját. Jegyzetet készítsünk a későbbi könnyebb visszaemlékezés céljából.

- Alapesetben nem kérdezőnk (csak utólag). Megegyezés esetén lehet kérdezni, de akkor is csak keveset. Leginkább akkor, ha a használónk közlékeny, szívesen beszél a munkájáról és láthatóan semmiben sem akadályozzuk.

- Elsősorban a célok és feladatok megismerése, vagy éppen a megoldások módjának részletes megismerése. (Ez utóbbi nehezebben deríthető ki passzív megfigyeléssel.)

A napi rutin (rutinmozdulatok megfigyelése). Egy egyszerűbb probléma megoldási folyamatát lekövethetjük csak odafigyeléssel, de egy bonyolultabbát csak kérdezéssel.

- Az események rögzítése jegyzeteléssel (itt nem zavar, idő is több van rá). Mindent írjunk le, ami fontos lehet. (hova kattint, hányszor kattint, bosszankodik-e egy-egy funkció használatakor, mi lehet ilyenkor a problémája stb.)

- Időmérés, időpontok és időtartamok följegyzése. Egy feladat megoldásához mennyi időre van szüksége a használónak – ha túl hosszú időt vesz igénybe a megoldás megtalálása, akkor erre figyelniünk kell a mi programunk megírásakor. A funkció okozza a hibát? Vagy

ergonómiailag rosszul van elhelyezve valami a programban? Írjuk le a hiba lehetséges okát, majd, ha lehetőségünk adódik, kérdezzük meg a használó véleményét.

- Az értékelésnél segít a magnetofon, video-magnó felvétel (csak engedéllyel). Többletköltség, a felvétel átírása jegyzőkönyvbe nehézkes, költséges.

- „Kommentált műveletek”: a személyt megkérjük feladatának, észlelései, gondolkodása, döntései, érzelmei elmondására – magnetofon felvétel. Ezeket az információkat a legjobb a munkavégzés közben rögzíteni. Kérjük meg a használót, hogy mondja hangosan, hogy éppen mit csinál, és azt is, hogy hogyan lenne számára kényelmesebb az adott feladatot megoldani. Sok hasznos információt nyerhetünk ezzel a módszerrel.

5.1.5 Szerepátvétel

A feladatok és tevékenységek előzetes megismerése (a többi módszerrel) után a megfigyelő maga áll be a munkát végezni.

- Célszerűen egy gyakorlott személy mellé, hogy tudjunk segítséget kérni, ha elakadnánk egy feladat megoldása közben.
- Elsősorban egyszerűbb, könnyebben tanulható feladatokra, hiszen a bonyolult feladatokat csupán megfigyelés után nem tudjuk megoldani.
- Cél: a részletek megismerése, további tapasztalatok gyűjtése
- Jegyzőkönyv inkább csak utólag készíthető.
- A munka előzetes megismerése (más módszerekkel). Dokumentációk olvasása a használandó programról, a munkafolyamat-leírások áttanulmányozása.
- Eleinte segítség kell ez nem kétséges.
- Jellemző ismeretek a megtanulhatóság mértékéről is. Információt kaphatunk arról, hogy egy ilyen bonyolultságú program használatát mennyire nehéz elsajátítani.

6. Kezelőfelület tervezése, interakció stílusai

6.1 Az interakció stílusai

Az interakció stílusait, mint fogalmat a következőképpen fogalmazhatjuk meg:

„Az interakció folyamán, a képernyőn megjelenő információ, és a kezelő inputjainak, valamint ezek alapján a programfunkciók szervezési „stílusának” állandó, ötletszerű változása megnehezíti a kezelő munkáját. Jól használható programok kezelőfelülete általában hosszabb-rövidebb ideig egyfajta stílust követ, de - a munkafolyamat részfeladatai jellegének megfelelően - ezek időben cserélődhetnek.”

Napjainkban a legnépszerűbb interakciós technika a grafikus kezelőfelület.

Az interakciós stílusokat osztályozhatjuk, csoportosíthatjuk. Sokféle nézőpont létezik, de a ma ismert legjobb osztályozás Newton és Lemming könyvében lelhető fel.

Ez a következő:

- billentyűzetet használó, módváltó stílusok
 - menü alapú interakció (hierarchikus menük)
 - kérdés-felelet interakció
 - egyedi kezelőszervekkel (funkcionális gombokkal) irányított interakció
- közvetlen manipulációs stílusok
 - grafikus közvetlen manipuláció
 - űrlapkitöltés
- nyelv alapú stílusok
 - parancsnyelvű interakció
 - természetes nyelvi interakció

Ezt az osztályozást kiegészíthetjük program specifikus pontokkal, amelyekkel éppen dolgozunk.

6.2 Billentyűzetet használó, módváltó stílusok

6.2.1 Menü alapú interakció

„A menü az interakció szemléletes segédeszköze: a képernyőn megjelenített választékból rámutatással lehet választani. A választható lehetőségek szavak, rövid szövegek, vagy szimbolikus képek, ikonok formájában, egységes keretben jelennek meg a képernyőn és ezek valamelyikére egérrel, a tablet stílusával, vagy közvetlenül az újjunkkal mutathatunk rá.”

Menü használata esetén tehát rámutatással tudunk választani az elénk táruló lehetőségek közül, legyen az eszközünk akár az egér, vagy a mai modern technológiának köszönhetően az érintőképernyős felületeken ujjunk segítségével is kiválaszthatjuk a megfelelő funkciót.

A menü használata nagyon kényelmes ugyan, de ha csak ezt a fajta stílust alkalmazzuk, akkor a felhasználó egy idő után irányítottnak érzi magát, unalmassá válik számára a program használata.

Figyelnünk kell arra, hogy ha egy menü funkció nem elérhető, jelezzük ezt kiszürkítéssel vagy más egyértelmű módon a felhasználónak, hogy tudja, nem az ő hibájából nem működik a funkció.

A menük előnyei: használatuk gyors, egyszerű, kényelmes, a menük használata könnyen tanulható, eseti használóknak megfelelő, a menü látványa tájékoztatás a lehetőségekről.

A menük hátrányai: folyamatos használatuk hosszú időn át unalmassá válik, a hierarchikus menüknél a gép irányítja a műveleteket, a mélyebben lévő elemeinek elérése lassú, időigényes, bonyolult struktúra alakulhat ki.

A menü lehet:

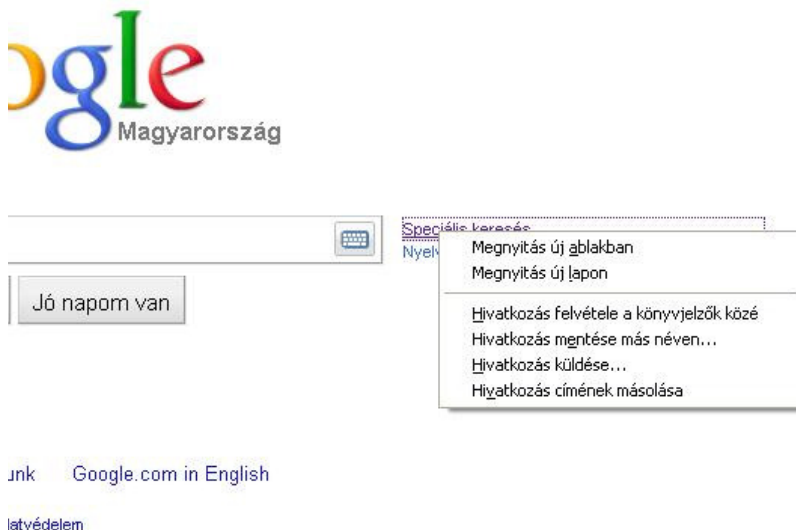
- sormenü - más szóval lécmenü (általában a programok tetején a menüsor)
- oszlopmenü (például jobbklikkre előbukkanó menü)
- táblázatos menü (legördülő menü)

A menüket úgy kell elhelyezni, hogy ne takarjanak el értékes információt a kezelőfelületen, valamint az állandó menük kis helyet foglaljanak (a képernyő 4 szélén helyezük el őket). A

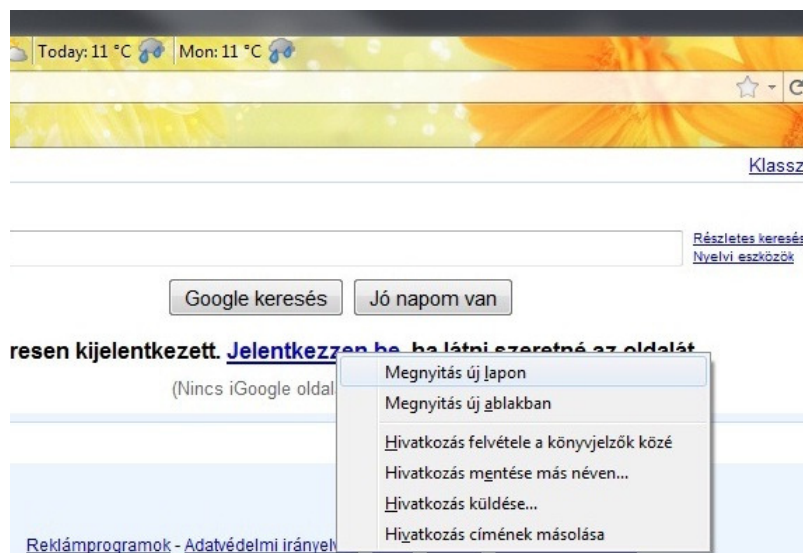
felugró menük lehetőleg minél feltűnőbbben bukkanjának elő, mutatva azt, hogy innen lehet elérni a kívánt funkciót, viszont helyezzük el úgy őket, hogy ne takarjanak el értékes információt a felbukkanásukkal. Az ideiglenes menüket vagy eszköztárakat tervezzük mobilisra, egérrel (ujjal) áthelyezhető legyen a képernyő más pontjára, tetszőlegesen megnyitható és bezárható legyen.

Figyeljünk arra, hogy, ha a menüben az elemek sorrendjét megváltoztatjuk, akkor még a gyakorlott felhasználókat is megzavarhatjuk. Vegyük példának a böngészők felugró menüjét. Ha az egér jobb gombjával egy linkre kattintunk, akkor megjelennek a „megnyitás új lapon”, „megnyitás új ablakban lehetőségek.” Különböző böngészőkben ezek különböző sorrendben szerepelnek. Engem, mint gyakorlott Mozilla Firefox használatát megzavart, amikor ezen menü elemeit felcserélték a fejlesztők az új verzióban.

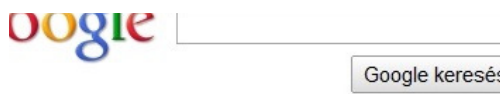
Menü a Firefox3-ban



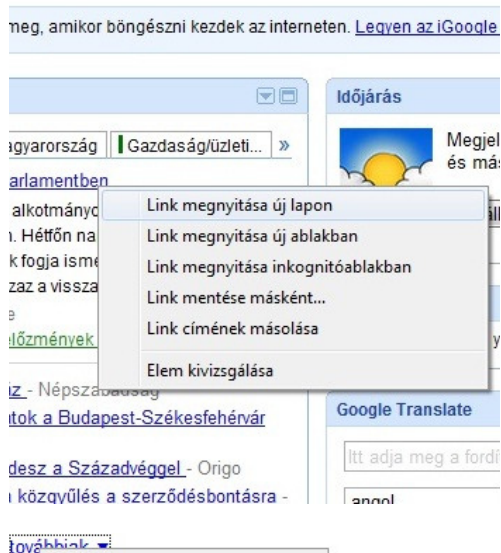
Menü a Firefox4-ben



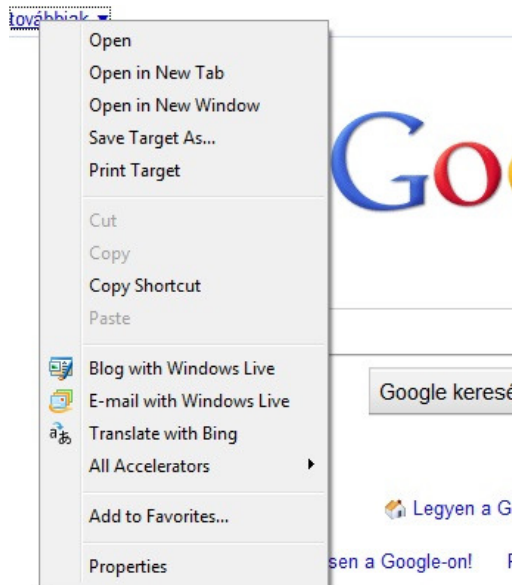
Kíváncsi voltam, hogy a többi böngészőben, hogyan néz ki ez a menü.



Menü Google Chrome-ban



Menü Internet Explorer-ben



Amint látható a leggyakrabban használt böngészők legújabb verzióiban az új lapon való megnyitás szerepel feljebb a menüben. Mindenesetre, az ilyen aprónak tűnő változtatások is kellemetlenséget okozhatnak a használóknak.

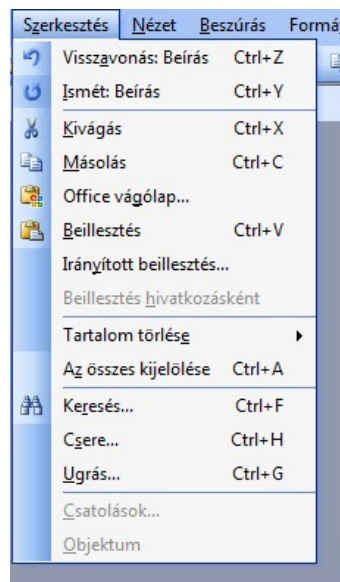
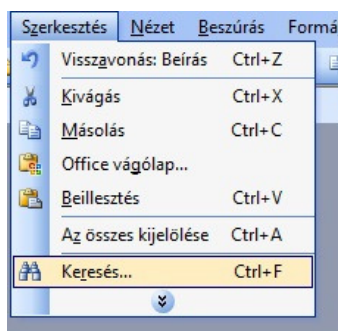
A menü lehet parancsmenü vagy paraméter menü, attól függően, hogy egy funkciót indítunk el a segítségével vagy paramétereket állítunk be. A menük elnevezése sokban segíthet a felhasználóknak abban, hogy eldöntsék, milyen típusú menüvel van dolguk. Címezzük a menüt igével ha parancsmenü és főnévvel ha paraméter menü.

Például nevezzük a menüt „nyomtat”-nak vagy „nyomtatási kép”-nek. Ebből tudni lehet, hogy az első a nyomtatást, mint funkciót indítja, a második a nyomtatás paramétereit finomíthatja.

Sokban segítheti a felhasználókat az is, ha a menü neve után teszünk három pontot. Például „Nyomtatás...”. Ez a jelölés azt sugallja, hogy mielőtt a funkció elindulna további beállítási lehetőségeink lesznek, nem rögtön indul el a nyomtatás.

A menük szövegezésekor kerüljük a rövidítéseket a könnyebb érthetőség kedvéért.

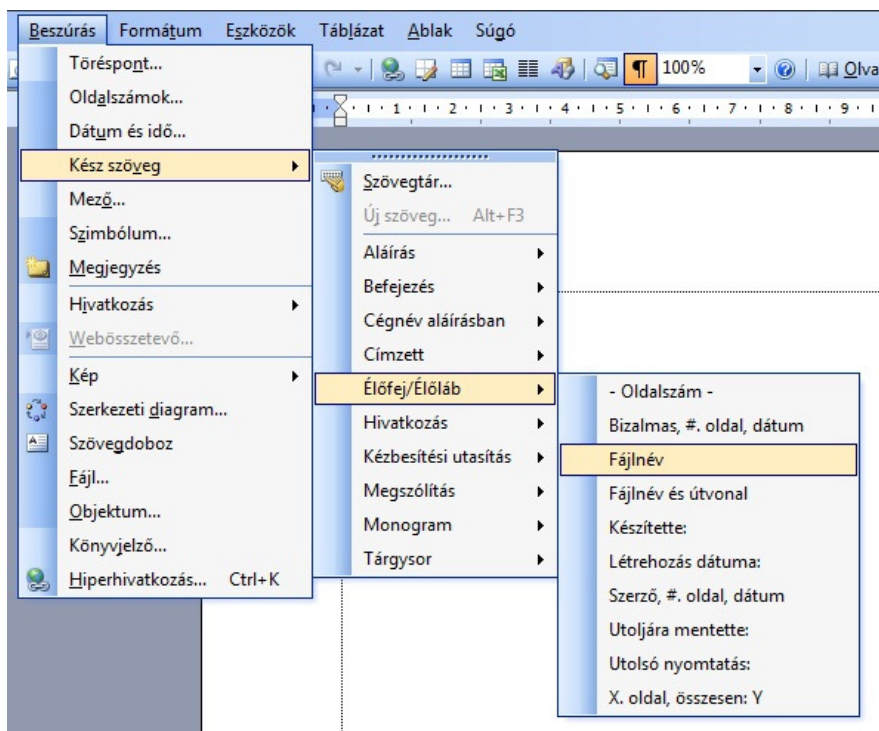
Táblázatos menük esetén egy listából tudunk választani. Ennek előnye, hogy sok elemet egy helyen csoportosítva tudunk megjeleníteni, hátránya, hogy bonyolultabb, nehezíti a helyes funkció kiválasztását. Egy táblázatos menü legördülő részében nem érdemes 7 elemnél többet megjeleníteni, mert az emberi agy ennél több információval nem könnyen boldogul. Érdekes megjegyezni, hogy abban az esetben, ha 7-nél több funkció van a menüben az új programok egyszerre mégis csak 7-et jelenítenek meg. A többi elérhető, ha a menü alján a kis nyílra kattintunk. (Például a Microsoft Word szövegszerkesztőjének menüje ezen az elven működik.)



Hierarchikus menüről beszélünk, ha a legördülő menü egyes elemei további menüket nyitnak meg. Tipikusan akkor van erre szükség, amikor túl sok funkciót szeretnénk elérhetővé tenni a menüből, de túlszűfolt lenne a menü, ha

az eddig tárgyalt menütípusokat alkalmaznánk.

A hierarchikus menü mélységét célszerű háromnál nem mélyebbnek választani, mert túl bonyolulttá válhat és a felhasználó összezavarodhat a funkciók keresése és kiválasztása közben.



A menüből a következőképpen választhatunk:

- érintésre érzékeny képernyőbevonat esetén újunkkal rámutatva,
- ennek hiányában a mutató rávezetése egérrel, vagy a tablet stílusával,
- vagy a billentyűzet nyíl gombjaival,
- a menüpontok szövegének kezdőbetűjével, vagy más kiemelt betűjével,
- lerövidített választással.

A tervezett programtól függően ajánlott többféle lehetőséget kínálni a felhasználóknak arra, hogy elindítsanak egy funkciót. A gyakran használt funkciókat a billentyűzet segítségével (billentyűkombinációval) is el lehessen indítani, valamint kattintással is működjön. Különbséget lehet tenni az egyes funkciók között azzal is, hogy egy vagy két kattintással

indulnak-e. Azok, amelyek két kattintást igényelnek lényeges változást idézhetnek elő, amelyek csak egy kattintást igényelnek, azok egyszerű paraméter beállítások, vagy nem nagy hatást kiváltó műveletek.

Menük használatakor fontos az, hogy a program visszajelzést adjon arról, hogy a menü melyik részén járunk, az egerünk éppen melyik funkcióra mutat. Ez jelölhetjük könnyen színezéssel (aktív menüpont színe üssön el a többi menüpont színétől), valamit hierarchikus menük esetén a hierarchia felépítése végig látható legyen, amíg elérünk a kívánt funkcióhoz. (lásd előző kép). A visszajelzés egy formája lehet, ha a menü elemekre egerrel ráállva, kis felugró ablakban, rövid információ jelenik meg az adott funkcióról. Ezt az opciót ki/bekapcsolhatónak célszerű tervezni, ugyanis zavarhatja a felhasználók egy csoportját.

A menük szövegezésénél próbáljunk törekedni arra, hogy minél inkább kerüljük a számítástechnikai szakkifejezéseket, minél beszédesebb menü elnevezéseket használjunk. Ha egy konkrét szakterületre fejlesztjük a programunkat, akkor a terület szakszavait beépíthetjük a menünevekbe. Figyeljünk arra, hogy beszédes neveket adjunk a menün belül, ami egyértelműen utal a mögötte lévő funkcióra. Általánosan elfogadott, hogy a szavakat nagybetűvel kezdjük és csupa kisbetűvel folytatjuk a jobb olvashatóság érdekében.

Miután egyszer gondosan kialakítottunk a menünk szerkezetét (a sormenüben és a hierarchiában is), akkor törekedjünk arra, hogy a későbbiekben ezt a sorrendet és a menü funkciók logikai csoportosítását meghagyjuk. Ha a felhasználók már régóta használják a rendszert, akkor a funkciókat a megszokott helyükön keresik, és összezavarodhatnak, ha nem találják a funkciót a helyén. Abban az esetben, ha egy funkció nem elérhető (pl. ideiglenesen, de várhatóan visszakerül a menübe), akkor ne távolítsuk el a megszokott helyéről, jelöljük más színnel (tipikusan kiszürkítéssel) a funkció inaktivitását.

A legfontosabb tervezési szempontok tehát:

- A főmenü legyen könnyen elérhető
- Legyenek rövidebb utak
- A menüelemek szövege legyen egyértelmű, és jól megkülönböztethető
- A címek legyenek összhangban a menü szövegekkel
- Ügyelj a menüelemek sorrendjére

- Hosszabb gépelésnél lehessen előreszaladni
- A címek legyenek összhangban a menü szövegekkel
- Ügyelj a menüelemek sorrendjére
- Csak az aktív menüelemeket mutasd, illetve a többit halványítsd
- Menüszerkezet legyen logikus felépítésű!
- A menüpontok legyenek logikai sorrendben!
- Egy menüpont csak egy menüben szerepeljen. Ha több helyen szerepel, az megzavarhatja a kezelőt, és nehezíti a betanulást
- A menük és menüpontok elnevezésében lehetőleg kövessük a hagyományokat. Egy menüpontot a neki hagyományosan megfelelő menübe helyezzünk
- A kiválasztás módja legyen átgondolt!
- Biztosítsuk a felhasználónak a visszalépést az előző menüszintre!
- Minden képernyőn tájékoztassuk a felhasználót, hogyan tud visszalépni, kilépni, előre jutni!
- Minden helyzetben tudja a felhasználó, hol van!
- Az elrendezés és a terminológia legyen következetes!
- Biztosítsunk rövidítéseket!

6.2.2 Kérdés-felelet interakció

A kérdés-felelet interakció lényege, hogy a számítógép szöveges kérdést tesz fel a használnak, aki szintén szöveggel válaszol. A legtipikusabb példa erre a login/password. Ez az interakció jó, mivel nem igényel tanulást és gyorsan működik, kevés funkció esetén. Viszont sok és bonyolult funkciók esetén nagy a hibázás lehetősége a szöveges válaszok miatt. A felhasználó komfortérzete csökkenhet amiatt, hogy a számítógép irányít.

The image shows a login form with a blue background. It contains two input fields: 'E-mail cím' with the text 'probauser@gmail.com' and 'Jelszó' with a masked password '.....'. Below the email field is a checkbox labeled 'Emlékezz rám'. Below the password field is a link 'Elfelejtetted a jelszavad?'. To the right of the password field is a blue button labeled 'Bejelentkezés'.

Példa kérdés-felelet interakcióra

6.2.3 Egyedi kezelő szervekkel irányított interakció

Tipikusan az automatákra, a nyomógombokkal működő berendezések használatára kell gondolni ennél az interakciónál. Minden egyes lépés egy konkrét állapotba viszi át a használót. (liftet megnyomja az 5ös gombot, akkor a lift elindul az 5. emeletre). A használata egyszerű kevés választási lehetőségnél, sok nyomógombos, bonyolult funkciók kiválasztásakor lassú lehet, nehézkes a kezelés.

6.3 Közvetlen manipulációs stílusok

„A közvetlen (direkt) manipuláció interakciós stílus, amelynél az ember az egérrel végrehajtott egyszerű műveletekkel építi a munka képernyőn látható tárgyát. A képnek és a műveleteknek a valósággal való szoros, szemléletes és műveleti hasonlósága miatt a folyamatos munka során „olyan mintha”; az embernek az érzése mintha műveleteivel közvetlenül maga módosítaná a munka tárgyát.”

6.3.1 Grafikus közvetlen manipuláció

A grafikus közvetlen manipuláció legfontosabb feltételei:

- a munka tárgyának folyamatos látványa a képernyőn
- gyors, egyszerű, növekményes, megfordítható, mozgásos műveletek
- amelyeknek azonnal látható a hatása
- szoros szemléletes és műveleti hasonlóság (analógia) a munka tárgyával,
- folyamatosan végzett munka

Jellemzi:

- az “olyan mintha” érzés
- gyors és könnyű, réteges vagy spirális tanulást tesz lehetővé
- készségek, hosszan megtartható és könnyen újra fölidézhető

- gyors munkavégzés
- bátorítja a kísérletezést
- sikerélmény, szubjektív érintettség, élvezet.

„A szoros szemléletes hasonlóság: a munka tárgya, és annak képe a képernyőn; amit a kezelő jól ismer. (A szakembernek egy elektronikai kapcsolási rajzzal is lehet közvetlen manipulációs érzése.) Műveleti hasonlóság: a műveletek végrehajtása is valamilyen módon emlékeztet a munka tárgyán végrehajtani kívánt műveletekre. (Nem jó a szemétkosár ikonját a képernyő tetejére tenni.) Ezek együtt alkotják a „szcenáriót”. „

Optimális esetben a munka tárgya teljesen kitölti a képernyőt, minden más csak elterelné a használó figyelmét a munkafolyamatról.

„ A scenárió szemléletes értelmezése legyen *egyszerű*: mind az ábraelemek, mind ezek viszonyának és változásinak értelmezése. Kevésféle, jól fölismerhető képelem, a kép könnyen fölismerhető szerkezete, 2-3 csoportosítási szint, 2-3 vonaltípus és elemméret. Az elemek viszonylagos helyzetének - közelség, összekötés, alá-fölé, előtte-utána - egyszerű jelentése, vagy ilyen jelentés hiánya. „

Törekedjünk az egyszerűsége és egyértelműsége. Minél inkább sikerül ezeknek a feltételeknek eleget tennünk, annál jobban használható lesz a programunk.

A műveletek végrehajtásánál nagyon fontos, hogy a végrehajtás után azonnal lássuk a művelet eredményét. (elemek elmozdítása, színek változtatása). Ha a visszavonás művelet könnyen végrehajtható, akkor növelhetjük a kezdeményezési, próbálgatási kedvet, lehetőséget teremtve ezzel a könnyebb tanulhatóságra. Ebben az esetben nagyon zavaró lehetne felugró hibaüzeneteket alkalmazni. Leginkább egyáltalán nem alkalmaznak hibaüzenetet. Ha végrehajtottad a műveletet megjelenik az eredmény. Lehet, hogy nem a várt eredményt kapjuk. Viszont a művelet visszavonásával könnyen korrigálhatunk. Létezik a visszavonás ellentétes művelete, amit magyarul legkifejezőbben „mégis” műveletnek hívhatunk. Ez a visszavont műveletet állítja helyre, ha időközben rájöttünk arra, hogy mégis jó volt, amit csináltunk. Mindkét műveletre igaz, hogy egyes programok csak egy vissza és mégis műveletet tudnak végrehajtani (nem lehet a munka kezdetégi visszalépkedni), ez a szám természetesen definiálható az egyes programok esetén (általában 5 művelet vonható vissza).

Az ilyen típusú programok kezeléséhez nem mindegy milyen eszközöket használunk. Érintőképernyős alkalmazás esetén csak az ujjunkra van szükség a kezeléshez (esetleg pálcára). Számítógépes kezelés esetén az egér a legelengedhetlenebb eszköz. Amennyire anyagi forrásaink engedik a legjobb minél nagyobb képernyőn dolgozni.

Az ilyen programok esetén hátrány elavult géppark vagy rossz minőségű eszközök használata. Kis képernyőn, aminek a felbontása is kicsi, rosszul lehet látni az ikonokat, elmosódhatnak az apróbb elemek. Régi típusú egér használata is megnehezíti a munkát. Ha nehezen kezelhető az eszköz, akkor a használó frusztrálttá válhat, nem lesz sikerélménye, szenvedéssé válhat a program használata.

6.3.2 Űrlapkitöltés

Az űrlap az egyik legegyszerűbben használható és legkönnyebben tanulható interakciós stílus. Nem igényel sok számítástechnikai előképzettséget, főleg, ha úgy terveztük meg az űrlapot, hogy az előtte használt papír alapú megfelelőjére nagyban hasonlít.

Mivel a papír alapú űrlap megfelelőjét próbáljuk számítógépen megjeleníteni, ezért segítség képen szöveges segítségeket is feltüntethetünk az űrlapon, vagy választhatjuk azt a lehetőséget is, hogy klikkeléssel előhívható a segítség, így nem foglalja a helyet az űrlapon.

Ha hosszabb űrlapokat akarunk készíteni, akkor nem fog az egész űrlap egyben kiférni a képernyőre, azért a lapozást is meg kell valósítanunk.

Az űrlap kitöltendő rovatai előtt rendszerint egy kérdés helyezkedik el, amire lehetőségünk lehet két vagy több válaszból egyet megjelölni, szöveges vagy számok részére mezőket elhelyezni.

A mezőket célszerű úgy csoportosítani, hogy a kérdések logikai sorrendje megfelelő legyen, a hasonló témájú kérdések egymást követően legyenek feltéve.

A mezők közötti váltáshoz nagyobb szabadságot adhatunk a használónak, attól függően, hogy a billentyűzettel vagy az egérrel szeret jobban dolgozni. A billentyűzeten legcélszerűbb a tabulátor gombot használni a mezők közötti váltáshoz, ez általánosan a programok nagy részében lehetőségként táruul a használók elé, valamint megengedhetjük, hogy az egér segítségével, aktintással váltsanak a mezők között.

Személyes adatok

Családi név*:

Keresztnév*:

Születési dátum*:

Állampolgárság*:

Családi állapot*:

Elérhetőségek

E-mail cím*:

Telefon*:

Megye:

Város*:

Utca, házszám*:

Tanulmányok

Iskola neve*:

Végzettség foka*:

Szak megnevezése*:

Befejezés éve*:

Iskola neve:

Végzettség foka:

Szak megnevezése:

Befejezés éve:

Példa űrlapra

6.3.2.1 Tervezési szempontok

Első nagyon fontos szempont, a már említett hasonlóság a papír alapú űrlap és a programunkban megjelenített űrlap között, főleg abban az esetben, ha a leendő használók eddig még nem használtak számítógépes programot a munkájukhoz.

Nyújtunk minél több információt és segítséget a használatnak a kitöltéshez. Ez a legegyszerűbb módon az F1 billentyű lenyomásával elérhető, könnyen megjegyezhető és az űrlap fejlécében vagy láblécében is feltüntethető.

Egyértelműen címkézzük az egyes mezőket, látható legyen, hogy melyik szövegmező melyik címkéhez tartozik. A mezők kerete legyen megerősítve a jobb láthatóság kedvéért. A mezők háttérszíne jelezze, hogy melyik az aktuális, aktív mező. Ajánlaton két eltérő színt alkalmazni,

olyanokat, amelyeknek van halványabb árnyalata. Például az összes mező háttere lehet halványsárga, az aktuális mező pedig halványrózsaszín.

A hibajavítást mindenképpen engedélyezzük, mert az elgépelés lehetősége főleg gyakorlatlan felhasználó esetén eléggé gyakori. Elgépelés, vagy nem megfelelően kitöltött szövegmező esetén adjunk jelzést (hangjelzés, felugró ablak, szöveges üzenet), amelyben jelezzük a hibát, és megoldást kínálunk a javításra (a megfelelő formátum megadásával, vagy csak szöveget fogad el a mező, csak számot fogad el a mező stb.)

Ha a felhasználó kitöltötte az űrlapot, legyen lehetősége az ellenőrzésre. Lehessen újra végigmenni a beírt adatokon, több oldalas űrlap esetén is. Készítsünk egy mentés/elfogadás/küldés gombot, amelynek lenyomásával egy utolsó ellenőrzés fut le a beírt adatokon, ami újra ellenőrzi a formátumokat, vagy, hogy nem maradt üresen kötelezően kitöltendő mező. Sikeres ellenőrzés után jelezzük az eredményt elküldve/mentve/elfogadva szöveggel, valamint, ha hibát találtunk, akkor adjuk meg a pontos helyét, a mező nevét, amit javítani kell a könnyebb megtalálhatóság miatt.

Mindezeket betartva felhasználó-biztos űrlapot készíthetünk.

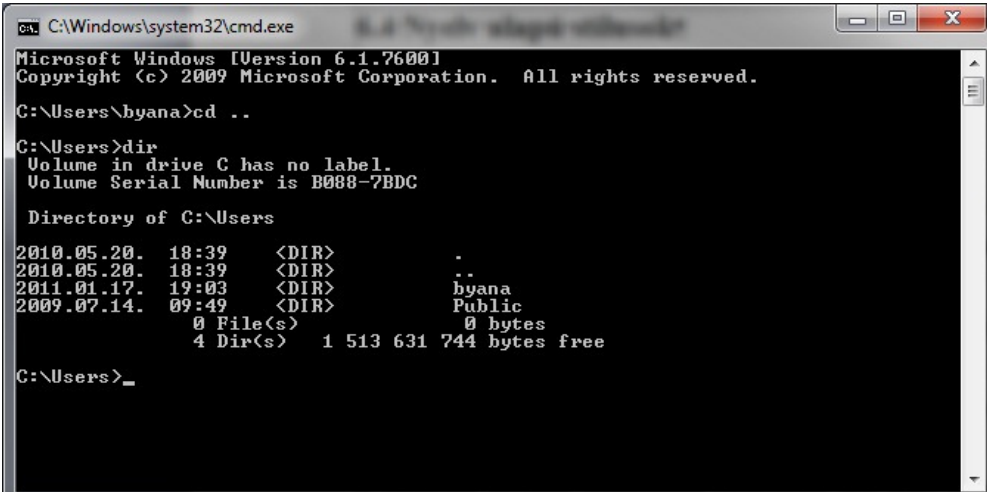
6.4 Nyelv alapú stílusok

6.4.1 Parancsnyelvű interakció

A parancsnyelvű interakció volt a legelső kialakult interakciós stílus. Kezdetben nem állt rendelkezésre grafikus felület, csupán a parancssor, ahová szabályosan begépeltek utasítások sorozatával tudtunk kommunikálni a programokkal.

Nehézsége egyben előnye is. Csak meghatározott parancsok, jól definiált szintaktikával, kapcsolókkal és paraméterekkel működnek. Ezek elsajátítása sok időbe telik, nagy az elgépelés veszélye, nehézkes a használata. Viszont éppen szabályszerűsége miatt közkedvelt használók egyes csoportjainál.

A parancsok neveinek megadásánál törekedjünk arra, hogy beszédes neveket adjunk, ez is segíti a könnyebb tanulhatóságot. Ha egy parancsot sok opcióval meg lehet hívni, akkor készítsünk listát ezekről a lehetőségekről és például egy „help” parancs kiadásával legyen kilistázható a parancsok használhatóságának összes módja.



```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\byana>cd ..

C:\Users>dir
Volume in drive C has no label.
Volume Serial Number is B088-7BDC

Directory of C:\Users

2010.05.20.  18:39    <DIR>          .
2010.05.20.  18:39    <DIR>          ..
2011.01.17.  19:03    <DIR>          byana
2009.07.14.   09:49    <DIR>          Public
               0 File(s)                0 bytes
               4 Dir(s)      1 513 631 744 bytes free

C:\Users>_
```

Példa
parancsnyelvi
interakcióra

6.4.2 Természetes nyelvi interakció

„Verbális interakció, kérdéseket a képernyőn, a válaszokat billentyűzetten keresztül lehet megadni, illetve mindkettőt lehet beszéddel, hangszórón és mikrofonon keresztül. Ha a kezelő inputja rövid válaszokra szorítkozik, inkább Kérdés-felelet stílusról beszélhetünk; ennek megvalósítás lényegesen egyszerűbb.”

Előnyei:

- kötetlen szintaxis, kötetlen sorrend
- kevés tanulással elsajátítható az ilyen program használata
- nem lehet elfelejteni a kezelését, hiszen hangunkkal irányíthatjuk
- Kényelmes, mert nem igényel sok gépelés, kattintást, a hang áll a középpontban.

Hátrányai:

- költséges, ki kell építeni a megfelelő gépparkot, amely már alkalmas ilyen programok futtatására
- gyakran tisztázó kérdések kellenek ezért az interakció hosszadalmas lehet
- nyelvi problémák előfordulhatnak

Ez az interakciós stílus a jövőben sokat fog fejlődni, hiszen egyre inkább a kényelem, az emberek kiszolgálása a cél, ez a stílus a kényelmet és az egyszerűséget hivatott megvalósítani.

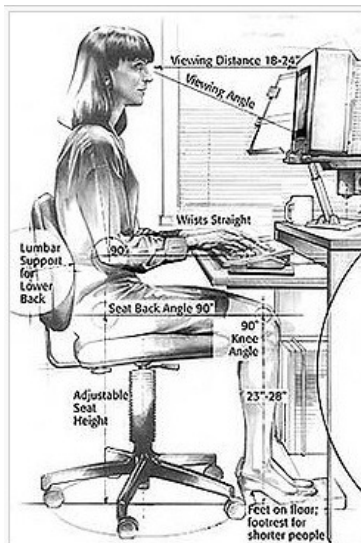
7. Munkakörnyezet ergonómiája

Az eddigi fejezetekben a szoftverek ergonómiájával foglalkoztam, viszont érdemes néhány gondolatot megfogalmazni a munkakörnyezet ergonómiájával kapcsolatban is.

Ahhoz, hogy a már ergonomikus szoftverünket hatékonyan tudják használni az emberek, elengedhetetlen, hogy mindezt kényelmesen, megfelelő irodai eszközök biztosítása mellett tehessék meg.

A munkaállomások különbözőek lehetnek attól függően, hogy ülő vagy állómunkát végzünk ott. Mindkét esetben fontos az elegendő mozgástér biztosítása, ne legyen a dolgozónak bezártság érzete, komfortosan otthonosan érezze magát.

Az egész napos ülőmunka egy kényelmetlen széken szenvedéssé változtathatja a munkát, azért fontos, hogy olyan székeket biztosítsunk a dolgozóknak, amelyeket testre lehet szabni. A helyes testtartás így néz ki:



A hát legyen egyenes, a szemünk (látásvonalunk) a monitor felső részére essen, a felkarunk és az alkarunk lehetőleg 90°-os szöget zárjon be, csakúgy, mint a testünk és a combunk és a combunk és a lábszárunk. Ez a testtartás terheli meg legkevésbé a gerincet. A helytelen testtartás következménye gerincferdülés vagy scoliosis kialakulása lehet. Ha túl közel ülünk a képernyőhöz, akkor romolhat a látásunk, a szemünk kiszárad, elfárad. Ha rossz szögben helyezkedik el a kezünk a billentyűzeten, akkor a csuklózületeink megfájdulhatnak.

Elkerülhető a szemünk romlása, kiszáradása, ha a munkáltatónk megfelelő minőségű monitorokat bocsát a rendelkezésünkre. Manapság már léteznek szemkímélő termékek, valamint létezik olyan bevonat szemüvegekre, amely kifejezetten a számítógép előtti munkára lett kifejlesztve.

Ezeket betartva nagyban csökkenhet a munkahelyi balesetek száma is. Jó minőségű felszerelések között a dolgozók jobban érzik magukat, nő a biztonság- és komfortérzetük. Ha

egyszer jól alakítanak ki egy munkaállomást, akkor a későbbiekben kevesebb juttatást kell biztosítani a dolgozóknak az egészségügyi ellátásukhoz, ezzel csökkentve a cég költségeit.

Az is fontos lehet, hogy engedjük meg a dolgozóknak, hogy személyesebbé alakítsák a munkakörnyezetüket. Néhány képpel, növénnel személyesebbé tehető a személyes szféra, így sokkal nagyobb kedvvel érkeznek munkába, nagyobb kedvvel dolgoznak, ami nagyobb hatékonyságot eredményez.

A munkaállomás ergonómiájára is igaz, csakúgy, mint a szoftverek ergonómiájára, hogy ha előre gondolkozunk, és előre megvalósítjuk a kényelmet jelentő teret és eszközöket a dolgozóknak, akkor ezzel pénzt spórolhatunk a későbbiekben és növeljük az elégedettséget ezzel a hatékonyságot is.

Befejezés

A dolgozat folyamán lépésről-lépésre ismertettem meg az olvasót a szoftver ergonómiai szemléletmóddal és a szoftvertervezés ergonómiai szempontjaival.

Igyekeztem bemutatni, hogy minden ergonómiai törekvés arra irányul, hogy az embereknek, a szoftver használóknak könnyebb legyen a végzendő munka. Minden, amit számítógép segítségével meg tudunk tenni, azért jött létre, hogy az ember munkáját segítse, azért is fontos, hogy ezek a programok és berendezések könnyen kezelhetőek, egyértelműen használhatóak legyenek, a lehető legkevesebb fejfájást okozzanak a használónak. Hiszen a gépek értünk vannak, emberekért, a mi munkánkat segítik gyorsabban, pontosabban, eredményesebben elvégezni.

Az ergonómia mindig többletköltséggel jár. Sokan emiatt inkább lemondanak róla, sajnálva a rá fordított időt és pénzt. Ez a költség viszont megtérül a későbbiekben. Gondoljunk csak bele, hogy a sok időt és pénzt felemésztő új szoftverünk nehezen használható és az ügyfél vagy nem veszi meg, vagy arra kér bennünket, hogy néhány dolgot változtassunk meg rajta. Már ez több költséggel járhat, mintha az elejétől fogva bevontuk volna a leendő használókat a tervezési folyamatokba, és a végén már csak apró változtatásokat kellene megtennünk. A fejlesztési folyamat során használó-interjúkkal, beszélgetésekkel fényt deríthetünk az igényekre és beépíthetjük azokat a fejlesztésbe.

Ha ezeket az ismereteinket összegyűjtjük, és írásos formában eltároljuk, akkor a következő alkalommal már tapasztaltabban foghatunk neki a szoftver ergonómiai tervezéséhez. Már tudni fogjuk, hogy milyen hibákat vétetünk legutóbb és mi lenne jó megoldás, hogy ne kövessük el ugyanazokat a hibákat. Ahogy bővül az irattárunk, és egyre nő a tapasztalatunk, úgy egyre csökken majd a költsége az ergonomikus szoftverfejlesztésünknek. Belátható tehát, hogy a jövőre fókuszálva érdemes belevágni az ergonómiai tervezés megismerésébe, hiszen erőfeszítéseink megtérülnek.

Úgy érzem, hogy a dolgozatommal elértem a célomat és egy csokorba gyűjtöttem a szoftver ergonómia legfontosabb elemeit, útmutatást adva ezzel azoknak, akik szeretnék ezzel a témával megismerkedni, és a későbbiekben ergonómiai szempontból is jó szoftvereket készíteni.

Irodalomjegyzék

William Newman: **What Application Designers Know: Some Thoughts on Innovation in Interactive Systems**

(Xerox Research Centre Europe Cambridge Laboratory, 1997 – Cambridge, UK)

Reinhard Oppermann: **User-interface design**

(Institute for Applied Information Technology, 2001 - Germany)

Scott Openshaw, Erin Taylor: **Ergonomics and Design**

(Allsteel Inc., 2006 – Muscatine, USA)

Eric M. Schaffer: **How to Develop an Effective GUI Standard**

(Human Factors International Inc., 2004 – Fairfield, USA)

Internetes tartalmak:

<http://krammer.web.elte.hu/infokar/szofterg/jegyzet/>

<http://en.wikipedia.org/wiki/Usability>

<http://en.wikipedia.org/wiki/Ergonomics>

<http://msdn.microsoft.com/en-us/library/ms997577.aspx>

<http://www.slideshare.net/meetoff/polgr-pter-balzs-hasznalhatsg-1x1>