Usability - použitelnost

Jak snadno se se systémem pracuje.

Převážně, ale nejenom, záležitost uživatelského rozhraní.

Hlavní zdroje

Nielsen, H., Usability Engineering, Academic Press, 1993

Mayhew, D.J., Principles and guidelines in Software User Interface Design, Prentice Hall, 1992

Faktory akceptovatelnosti IS (úspěchu)

Akceptovatelnost

- Sociální
- Praktická
  - *Technické vlastnosti* (cena, spolehlivost, robustnost, modifikovatelnost. Škálovatelnost, kompatibilita,…)
  - *Užitečnost.*
    - *Funkčnost (jaké funkce a služby nabízí)*
    - *Použitelnost (srozumitelnost ovládání, snadnost naučení, barvy na obrazovce), týká se hlavně uživatelského rozhraní*

Všechny faktory je třeba vyvážit

Faktory použitelnosti softwaru

- snadnost naučení,
- efektivnost při používání,
- dobře se pamatuje, jak systém používat,
- málo chyb uživatele způsobených špatným ovládáním rozhraní,
- subjektivní příjemnost práce se systémem pro uživatele,
- dobré ergonomické vlastnosti

Důležitost rozhraní

- Cosi jako obal systému, i obal (často hlavně) prodává
- Kvalitní rozhraní
  - ovlivňuje produktivitu až o 15%,
  - stres a tedy i spokojenost lidí (a ovlivňuje i další možné ztráty způsobené chybami, např. stresem),
  - snižuje množství chyb obsluhy (a jimi způsobemé ztráty
  - není programátorsky pracné, je pracné a náročné na rozmyšlení a testování

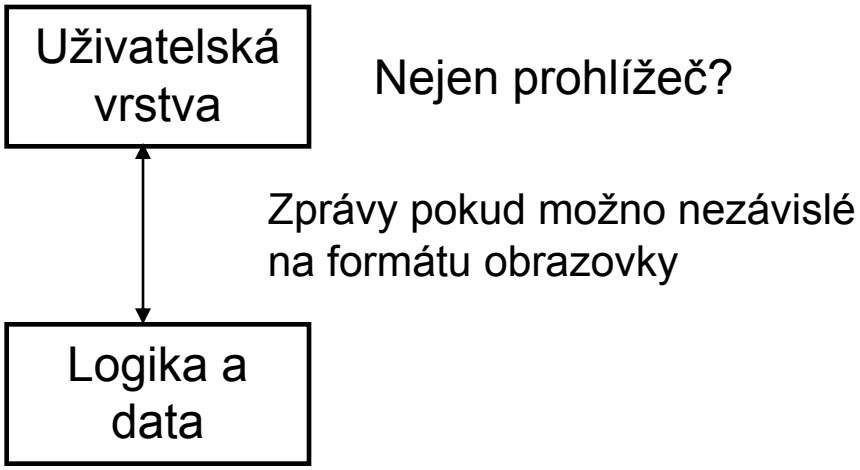
Specifika vývoje UI

1. Při vývoji UI je ještě obtížnější než při návrhu funkcí odhadnout optimální vlastnosti UI, protože ty závisí na psychologii, dovednostech a znalostech budoucích uživatelů.
2. Testování UI je možné pouze tak, že systém testují uživatelé. Testování se provádí sledováním práce uživatelů.
3. Vlastnosti uživatelů se během používání systému vlivem zkušeností mění.
4. UI musí vždy počítat se začátečníky i s pokročilými.
5. Důležitou roli hrají problémy ergonomie.

Specifika vývoje UI

- UI musí být navrženo v úzké spolupráci s uživateli. Návrhář UI většinou nedokáže dostatečně přesně odhadnout psychologii uživatelů, co jim bude vyhovovat a co nikoliv.
- Na druhé straně pro UI platí, že je nedokáže dobře navrhnout uživatel sám.
- Řešení tohoto dilematu je v prototypovém řešení UI vycházejícím z vlastností UI úspěšných systémů.

Uživatelská vrstva



Formát obrazovky závisí na zkušenostech uživatele, módách a také na tom, jak vystihneme psychologii práce. Musíme jej iterativně zlepšovat

Hlavní zásady vývoje rozhraní

1. Považovat vývoj rozhraní za relativně autonomní problém
   - Vytvořit v systému autonomní komponentu/vrstvu pro implementaci rozhraní
2. Přesvědčit management i zúčastněné, že je rozhraní skutečně problém
   - Upozornit na nutnost postupného vývoje
3. Zajistit prostředky a účast lidí
4. Naplánovat etapy (před oživováním systému, během oživování, při provozu)
5. Použít pokud možno to co existuje a poučit se od úspěšných realizací

Prototyp a postupný vývoj

- Prototypové řešení by mělo být ověřováno těmi, kdo je budou skutečně používat. Tak např. skladník ověřuje práci s UI potřebným pro provoz skladu a ředitel rozhraní manažerské části IS. Případy, kdy o UI rozhodují výhradně šéfové, např. náměstci, nebo pouze informatici odpovědní za IS, nekončívají dobře. Ti se samozřejmě musí účastnit vývoje těch částí, které používají
- Na testování prototypového řešení navazuje testování UI na postupně oživovaném systému.
- Výsledky testů se používají jako podklad pro další zlepšování UI. Zlepšené verze je nutno opět testovat. Celý proces se několikrát opakuje (iterovaný vývoj), dokud nejsou vlastnosti UI vyhovující. Iterovaný vývoj je snáze realizovatelný, je-li UI realizován relativně samostatným modulem, který je koncepčně navrhován jako téměř samostatná aplikace. V SOA to může být autonomní služba

Zásady

Vývoj UI je silně ovlivňován následujícími skutečnostmi:

1. Při vývoji UI je ještě obtížnější než při návrhu funkcí odhadnout optimální vlastnosti UI, protože ty závisí na psychologii, dovednostech a znalostech budoucích uživatelů. Vlastnosti uživatelů se navíc v průběhu času mění
2. Testování UI je možné pouze tak, že systém testují uživatelé. Testování se provádí sledováním práce uživatelů.
3. Vlastnosti uživatelů se během používání systému vlivem zkušeností mění (začátečník – pokročilý, módy).
4. UI musí vždy počítat se začátečníky i s pokročilými.
5. Důležitou roli hrají problémy ergonomie.

Obtížnost vývoje rozhraní

- Závisí na psychologii a znalostech uživatelů
- Souvisí s řadou oborů (ergonomie, výtvarné umění, psychologie)
- S časem se požadavky mění
  - Nové objevy a techniky (GUI, okna)
  - Různé typy aplikací (př. CAD)
  - Začátečníci a pokročilí
  - Všeobecný vývoj, rozvoj počítačové gramotnosti a v neposlední řadě módy
  - Technický pokrok (viz např. vynález myši, nověji hlasový vstup)
- Nutno ověřovat na lidech

Základní varianty rozhraní

- Dávkové – job control language (scripting)
- Znakové – jedna řádka (klasický UNIX)
- Obrazovkové, spíše znakové– editace obrazovky
- Grafické (GUI) vyžaduje myš.
  - Budeme se zabývat hlavně GUI. Znakové rozhraní je spíše pro vývojáře. V řadě směrů má větší sílu než GUI a někdy je vhodné i pro uživatele.

Obecné zásady tvorby rozhraní

- Nutno budovat postupně a mnohdy i za situace, že systém ještě nefunguje, je nutná flexibilita (dovednosti, a požadavky uživatelů se mění, mnohé nelze hned odhadnout předem)
- Balancovat účast uživatelů a vývojářů
  - Uživatel nemá zkušenosti a technickou intuici, nezná celý obor a často přeceňuje jednotlivosti. Zná ale své potřeby, dovede ocenit uživatelskou orientaci rozhraní
  - Vývojář má tendenci používat zkratky a stručné obrazovky (nevědomě předpokládá svoji úroveň znalostí)

Obecné zásady tvorby rozhraní

- Návrhu a testování rozhraní se účastní ti, kdo ho budou skutečně používat (nikoliv např. jejich šéfové), *týká se to i manažerů u těch funkcí, které budou používat. Musí najít čas a přistoupit na to, že „musí poslouchat“*
- Nespoléhat na dlouhé nápovědy vyžadující specifické akce (koncipovat jen jako záchranu v nouzi)
- Začínat od nejčastěji používaných částí
- Minimalizovat počet významem rozdílných údajů na obrazovkách na 7, je-li možné. Více než 12 je prakticky nepoužitelné
- Používat kaskády obrazovek (hierarchické menu)

Žádoucí vlastnosti rozhraní

Snadné k naučení

Dobře se pamatuje (pozor, není to totéž, co snadnost zvládnutí)

Efektivní při (častém) používání (klávesové zkratky, krátké texty)

Samovysvětlující (kaskádové menu)

Subjektivně příjemné

Chrání před chybami (chcete opravdu smazat soubory?)

Konsistentní (podobné věci na podobných místech s podobnými ikonami a podobným ovládáním)

Hlavní kroky tvorby rozhraní

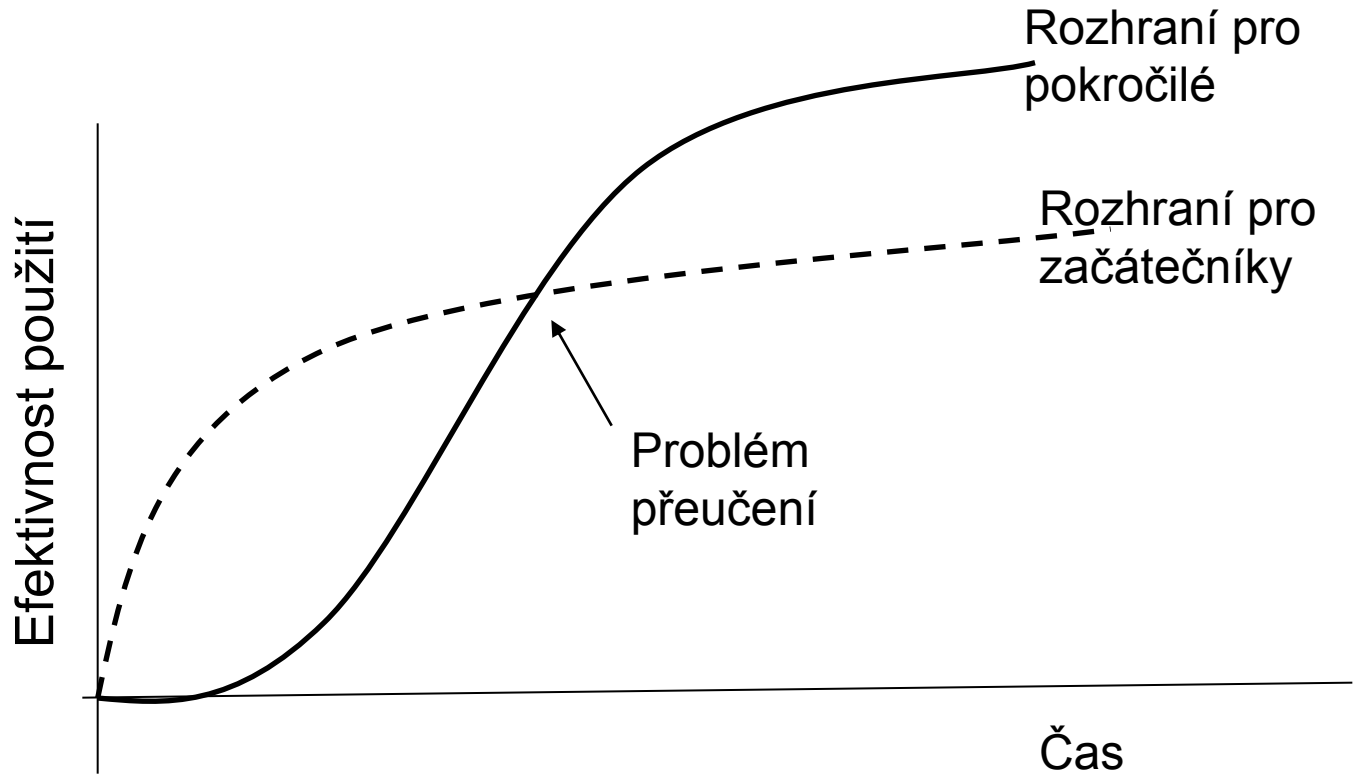
Uživatelské rozhraní (UI) je výhodné vyvíjet jako relativně nezávislý subsystém v obdobném životním cyklu jako celý systém.

Hlavní etapy vývoje UI jsou:

- Analýza požadavků, specifikace požadavků.
- Návrh UI.
- Realizace prototypů a jejich testování s uživateli.
- Integrace UI se zbytkem systému a testování UI společně s uživateli.
- Sledování vlastností UI za provozu a vylepšování vlastností UI.

Začátečník a pokročilý

- Začátečník preferuje snadnost naučení a snadnost pamatování, pokročilý spíše rychlost



Začátečník a pokročilý

- Začátečníci preferují GUI a učení metodou pokusů a omylů
- Je třeba najít cesty přechodu (přeučení) ke klávesovým zkratkám a znakovému rozhraní a jiným trikům vhodným pro pokročilé
  - Znakové rozhraní je mnohdy efektivnější a rychlejší

Kdo je začátečník

- Aspekty: Znalost oboru aplikace, znalost konkrétní aplikace a IT
  - Znalost oboru je svým způsobem rozhodující
- Vlastnosti konkrétního systému,
  - systém má být v harmonii s oborem aplikace
- Počítačová gramotnost.
  - Je výhodou je-li spojena s profesní zdatností a zájmem o zlepšování práce (viz zavádění systému),

Aspekty příjemnosti pro uživatele

- Pocit spolupráce, chová se jako dobrý a vstřícný spolupracovník
- Na obrazovce jen vše důležité, nic více, důležitější vlevo nahoře
- Pocit spolehlivosti (počítač se chová tak, jak očekávám (transparentně) resp. Nepřekvapuje a nechová se „nesrozumitelně“

Aspekty příjemnosti pro uživatele

- Vím, co se děje
- Jednoduchost a srozumitelnost z hlediska uživatele, intuitivnost
- Možnost kdykoliv vrátit akci a ukončit „proces“, to ale nelze vždycky, náprava je ale možná
- Přiměřeně rychlá odezva (do vteřiny, ne pár milisekund – může způsobit pocit „nestíhám“)
- Ergonomie, hygiena práce, střídat ovládání a činnosti

Jak zpřehlednit menu

- Problémem univerzálních systémů je, že nabízí příliš mnoho vstupních bodů naráz. Nepřehledné menu pro ně je hlavním problémem
  - Vybudovat strom prohledávání nízkého stupně větvení seskupováním funkcí podle příbuznosti
  - Budováním profilů uživatelů (co jsem dělal v poslední době)
  - Průvodci, příklady použití
  - Vyhledávání funkcí zadáním klíčových slov
- *Tento problém se silně podceňuje*

Životní cyklus budování rozhraní

1. Poznání uživatele
2. Analýza konkurenčních a existujících řešení
3. Rozbor a stanovení požadavků na přínosy kvalitního rozhraní
4. Návrh a implementace (paralelní návrh, syntéza, zlepšování)
5. Hodnocení prototypové implementace (včetně návrhu) za účasti koncových uživatelů, provádět i za provozu a pak spolupracovat s novými uživateli – staří nebudou ochotni se přeučovat)

Životní cyklus budování rozhraní

6. Společný návrh definitivního řešení
7. Heuristická analýza řešení (lze používat i v předchozích krocích
8. Prototypování (použití v obrazovkových prototypech)
9. Testování (s budoucími uživateli) na prototypech a při oživování systému
10. Iterativní vylepšování a rozšiřování
11. Sledování a úpravy za provozu.

Specifikace UI

A) Poznání potřeb uživatele, sledování jeho požadavků, ale také jeho práce, především scénářů a porocesů (viz obecné principy specifikace požadavků)

B) Analýza UI podobných nebo konkurenčních projektů Pokud existují podobné nebo konkurenční produkty, je výhodné prostudovat jejich UI, zjistit jejich přednosti a nedostatky.

Specifikace UI

C) Stanovení kontrolovatelných cílů. Některé vlastnosti UI je možné předem kvantifikovat. Typickým příkladem je počet chyb uživatele za jednotku času. Jako cíl je možné stanovit nějakou hodnotu této metriky. Hůře se formulují požadavky na snadnost osvojení a rychlost ovládání.

D) Stanovení finančního vlivu UI.

Poznání uživatele

- Osobnost a znalosti
  - Znalosti, zkušenosti, vzdělání, povaha
- Současné a v budoucnu plánované role
- Analýza činností, které provádí, hlavně scéháře a předávaná data
- Změny v profilu uživatele
  - (je přechod od začátečníka k rutinérovi možný?)

Poznání (koncového) uživatele

- Zajistit snadnost kontaktu, problémy
  - Dodavatel nechce, aby uživatelé znali vývojáře, bojí se, že by vývojáři byli zaměstnáváni operativou a konsultacemi
  - Snaha obchodníků držet co nejvíce ve své pravomoci
  - Chybí podpora od managementu uživatele
- Ověření znalostí a dovedností uživatelů
  - Zkušenosti, absolvovaná školení, osobní vlastnosti, schopnost spolupracovat

Poznání (koncového) uživatele

- Analýza a úprav funkcí
- Perspektivy uživatele a jeho systémů
- Očekávané změny ve znalostním profilu uživatele
  - jak dlouho bude zaměstnán,
  - má šanci být pokročilým uživatelem,
  - zvyšování kvalifikace

Návrh UI

A) **Paralelní návrh**. Při návrhu UI se osvědčuje nezávisle navrhnout více variant a pak vytvořit syntézu použitím toho nejlepšího ze všech návrhů.

B) **Spoluúčast zákazníka**. Spoluúčast je nutná při návrhu prvních verzí UI, nejlépe formou společného vývoje s tím, že první návrh formuluje vývojář.

C) **Koordinace návrhu**. Cílem je sjednotit formu UI pro různé funkce systému a zkombinovat dobré náměty.

D) **Heuristická evaluace** (viz níže, někdy i při specifikaci požadavků). Cílem je zjistit a napravit nedostatky uplatněním osvědčených zásad.

Paralelní návrh

- Rozhraní navrhuje a zčásti realizuje několik skupin. Předvedou návrhy zákazníkům
- Návrhy se vyhodnotí a použijí se nejlepší prvky ze všech návrhů
- Používá se u zvláště kritických systémů
- Neosvědčuje se návrh zákazníkem (srv. Specifikaci požadavků)

Heuristická evaluace

- Seznam dobrých rad pro vývoj uživatelského rozhraní a pro zlepšení použitelnosti. Existují seznamy až s několika stovkami rad. Hlavních deset doporučení následuje

Heuristická evaluace

1. Jednoduchý a přehledný dialog,
   - důležité vlevo nahoře, jen to, co je opravdu třeba
2. Srozumitelnost dialogu: dialog má být v jazyce a v souhlasu se zvyky uživatele (business oriented)
3. Malé nároky na paměť
   - bezstavová komunikace – nemusím si pamatovat, co bylo na předchozích obrazovkách
4. Konzistentnost
   - Podobné věci podobně na obdobném místě
5. Zpětná vazba
   - Uživatel vždy vidí, že se něco děje a jak to pokračuje

Heuristická evaluace

6. Jasně vymezené a snadno proveditelné exity.
   – možnost ukončit, nebo se vrátit o krok zpět
7. Klávesové zkratky pro často prováděné operace,
8. Kvalitní samy sebe vysvětlující hlášení chyb s odkazy na podrobnější vysvětlení
9. Snažit se vyloučit situace vedoucí k chybám
10. Help
    - Různé úrovně podrobnosti
    - I na papíře, text s odkazy

Heuristická evaluace

Analyzovat existující systémy (např. Microsoft). Je vhodná podobnost s těmito systémy, připouští-li to funkčnost (srv. grafické systémy). Lidé se UI snáze naučí

Jednoduchý a přirozený dialog v jazyce uživatele,

- Jen to, co je v dané etapě nepostradatelné, podoba s mezilidským dialogem

Minimalizovat nároky na paměť (bezstavovost)

Heuristická evaluace

Konzistentnost (stejné texty, barvy obrázky)

Zpětná vazba (teploměry, zprávy o postupu práce)

Jasně vymezené exity (ukončení, návrat o několik kroků,)

Zkratky (horké klávesy), ikony i menu pro časté akce

Kvalitní chybové zprávy (srozumitelný název a odkaz na podrobnosti)

Vylučovat situace vedoucí k chybám

Různé varianty nápovědy

Heuristická evaluace – výtvarné aspekty

Na obrazovce nemá být mnoho logicky odlišných údajů (do 12)

Nejdůležitější vlevo nahoře, střed

Nehýřit barvami a tvary, zatěžuje zrak a unavuje to mozek

Různé displeje nemají stejné podání barev a to může významně zhoršit čitelnost (nedávat spektrálně blízké barvy do popředí a do pozadí)

Heuristická evaluace – výtvarné aspekty

Teplé barvy vpředu, studené v pozadí

Při zdobnosti snáze uděláme chybný návrh barev

V provozu je důležité, jak je to pracné a jak mne to chrání před chybami

Zdobnost je vítána, je-li svým způsobem nápovědou a snižuje-li námahu

Zpětná vazba

- Reakce systému se pociťuje jako okamžitá je-li do desetiny sekundy. U psaní musí být ještě kratší
- Reakce okolo sekundy je tolerována, jde-li o složitější akci
- Reakce přes více sekund vyžaduje indikaci průběhu a nemá být příliš často

Cesty k poznání uživatele

- Snadnost kontaktu
  - Management nezajistil podmínky
  - Snaha obchodníků o udržení monopolu na kontakty
  - Obavy m-tu dodavatele, že budou vývojáři příliš zaneprázdňováni
- Vypracovat odhad finančních efektů na práci koncových uživatelů. Pozor na nepřímé úspory (často pro odhad nepřímých úspor chybí kvalitní data)
- Pozorováním lidí při práci odvodit ucelené činnosti a jejich scénáře a detekovat výjimečné situace, ověřit diskusí a předvedením obrazovek uživateli (často to musí vidět, aby to byl schopen posoudit)

Zpřesnění business procesů

Zjištěné činnosti zkusit provést sám a pak se zákazníkem

Sledovat, jak je činnost pokryta stávajícícm systémem (i ručně) a jak to řeší konkurenční produkty.

Stanovení kvantitativních cílů

- Počty chyb uživatele
- Doby provedení

Implementace a testování

A) Realizace prototypů

B) Předvedení prototypů UI.Uživatelé společně s vývojáři zkušebně používají prototypy dialogů typu Potěmkin.

C) Postupné vylepšování návrhu a odstraňování opomenutí a nedostatků.UI je zkušebně používáno uživateli a postupně se vylepšuje.

D) Vlastní testování

Testování rozhraní

Předvedení a zkoušení prototypů několika uživateli

- Někdy se vyplatí prvý test dělat s „brigádníky“ To je zvláště vhodné pro ověřování variant pro začátečníky
- Ne vždy lze, např. je-li nutná věcná znalost toho, co systém podporuje

- Log průběhu

Zahájení testu

Příprava nástrojů a lidí, místo, nástroje

Test provádí obvykle člen skupiny uživatelů za přítomnosti vývojáře.

- Je třeba brát ohled na velké individuální rozdíly mezi jednotlivými uživateli a také mezi nezkušenými a zkušenými pracovníky. Ověřování UI je proto třeba provádět s větší skupinou pečlivě vybraných budoucích uživatelů.
- Bývá výhodné provést nejprve zaškolení v ovládání systémových prostředků.
- Optimální počet testujících je 3 až 6.

Zahájení testu

Testéři pracují s tou částí UI, se kterou budou skutečně pracovat při provozu systému.

Při organizaci testů je žádoucí navodit atmosféru spolupráce:”Pracujete na tom, aby vám to, co budete používat, sloužilo dobře. Testuje se návrh, ne vy“.

Uživatelé nemají pracovat ve stresu.

Výsledky testů by měly být anonymní.

Upozornit, že se systém na základě požadavků testérů může změnit

Zahájení testu

Testování lze do jisté míry provádět na částečně funkčních prototypech. Vždy je však nutné nakonec provádět testy i na oživeném systému. Důvodem je potřeba testovat doby odezvy.

Je obvyklé, že se na základě analýzy výsledků testů UI podstatně mění. Testovat je nutné i nové verze UI.

U testování jen uživatelé, hlavně ne jejich šéfové

Body instruktáže při zahájení testu

- účelem testů je testovat systém, nikoliv uživatele;
- účelem testů je dosáhnout spokojenosti uživatelů;
- je žádoucí, aby se všichni svobodně vyjadřovali;
- poněvadž se jedná o testování, může výsledný systém fungovat poněkud jinak;
- poprosit, aby o testu testující mezi sebou nehovořili, aby tím ostatní testéry neovlivňovali;

Body instruktáže při zahájení testu

- zdůraznit, že účast na testu je dobrovolná a lze jej kdykoliv ukončit a že výsledky testu jsou anonymní a důvěrné;
- uvést, že jsou možné otázky, že je vítáno vyjádření pochybností, a že se má systém v budoucnu používat bez cizí pomoci;
- požádat o ”myšlení nahlas“, tj. poprosit, aby testující nahlas komentoval, co dělá a proč;
- vítány jsou pochvaly i kritické poznámky
- poprosit o pokud možno rychlou práci bez chyb.

Provedení testu

- Je důležité, aby testování nebylo přerušováno telefony, návštěvami atd.
- Je výhodné, když uživatel provede na počátku rychle a úspěšně nějakou část dialogu, je pak méně nervózní.
- Atmosféra při testování by měla být uvolněná.
- Nedávat najevo, že uživatel je pomalý nebo že dělá chyby.
- Vyloučit kibice, včetně (to především) šéfů testujícího.
- Zastavit testování, vznikne-li pocit, že testér toho má již dost.

Provedení testu, myšlení nahlas

- uživatel při testování „myslí nahlas“, říká
  - Co zrovna dělá a proč
  - Vyjadřuje překvapení, kritiku, nebo potěšení, případně že neví)
  - Problém je, že to zdržuje při práci, zvláště ty zručnější
  - Pokud uživatel říká, že neví nebo příliš dlouho váhá, lze mu pomoci (nemělo by být často, nutno zaznamenat)
- Je dobré na konec testování s konkrétní uživatelem se zeptat, co si o systému jako celku myslí

Vyhodnocení testu

- Požádat uživatele o celkový názor, především o to, jak je spokojen.
- Výsledky testů zaznamenat v dohodnuté formě.
- Provést analýzu žurnálu (ten je dodré sledovat i za (zkušebního) provozu
- Výsledky zavést do databáze projektu (pokud existuje).
- Sestavit vlastní hodnocení

Vyhodnocení testu

- Hodnocení problémů
  - 0 – celkem OK
  - 1 – kosmetická - opravit, bude-li čas
  - 2 - méně závažná – opravit,pokud nebouu závažnější
  - 3 – závažná – opravit co nejdříve, ale není překážkou pro zahájení provozu
  - 4 - katastrofická – systém nelze provozovat

Metriky pro hodnocení UI

Doba provedení určitého úkolu.

Počet variant úkolů úspěšně provedených za určitou dobu.

Poměr úspěšně provedených úkolů k počtu chyb.

Doba potřebná pro nápravu chyby.

Počet příkazů použitých během testů.

Počet příkazů, které nebyly vůbec použity.

Provoz

A) Sledování žurnálů (doby provedení, počty chyb)

B) Sledování názorů a změn chování

C) Pravidelné zaškolování

D) Úpravy podle zjištěných skutečností

E) Po větších úpravách nezapomenout na testy.

Evaluace- testování s uživateli

- Prototyp – Potěmkin
- Částečně funkční systém, hodnocení problémů
  - Jak vysoká závažnost
  - Jak velké omezení pro práci uživatele (včetně toho, jak často se objeví)
- Funkční systém:
  - Netestovat jen, zda správně pracuje ale také, zda jsou s ním schopni procovat uživatele → testovat více lidmi

Metriky pro hodnocení UI

Doba provedení určitého úkolu.

Počet variant úkolů úspěšně provedených za určitou dobu.

Poměr úspěšně provedených úkolů k počtu chyb.

Doba potřebná pro nápravu chyby.

Počet příkazů použitých během testů.

Počet příkazů, které nebyly vůbec použity.

Metriky pro hodnocení UI

Frekvence použití nápověd a manuálů.

Počty kritických a pochvalných komentářů uživatele.

Počet případů, kdy byl uživatel frustrován a kdy potěšen.

Rozsah mrtvých dob“, během nichž uživatel nekomunikuje.

Počet případů, kdy uživatel nevěděl jak dál.

Metoda	Etapa	Počet testérů	Hlavní výhody	Hlavní nevýhody
Heuristické evaluace	Návrh	0	Detekuje jednotlivé problémy použitelnosti, lze využít experty.	Není kontakt s uživateli.
Měření výkonů.	Testování UI, analýza podobných a konkurenčních produktů.	alespoň 10	Přesná data. Snadná kritéria porovnání.	Obvykle se nedaří porovnat všechny aspekty.
Myšlení nahlas.	Návrh, informativní evaluace.	3 až 5.	Levné. Zachycuje chyby v pochopení systému v poměrně reálné situaci.	Nepřirozené pro testujícího. Nevhodné pro zkušené. Ti rychleji jednají než mluví.
Pozorování.	Analýza a evaluace UI, tvorba scénářů.	alespoň 3 pro každý subsystém	Sleduje skutečné činnosti v reálu. Inspiruje návrh funkcí.	Subjektivní, obtížně kontrolovatelné, časově náročné, často se nenajde dost času na provedení.
Dotazníky	Analýza, sledování problémů za provozu.	Většina uživatelů	Anonymní. Lze opakovat, zachytí názory více uživatelů.	Nebezpečí, že důležití uživatelé neodpoví nebo dotazník přesně nepochopí.
Interview.	Analýza úkolů.	Několik	Flexibilní, lze jít do hloubky.	Časově náročné, náročné na kvality moderátora, subjektivní.
Žurnál (log)	Závěr testování, provoz.	Všichni uživatelé	Běží stále. Výhodné pro vyhodnocování efektivnosti a trendů.	Je nutno realizovat jednoduchý IS pro vyhodnocování.
Sledování názorů uživatelů.	Provoz.	Co nejvíce.	Lze sledovat trendy v názorech a potřebách uživatelů.	Obtížně se zajišťuje. „Proč se o to starat, když už to pracuje.“

Tab. 14.1: Přehled metod používaných při vývoji a modifikacích uživatelského rozhraní.

Hodnocení nedostatků rozhraní

Omezení uživatele při výskytu	Moc	Nevýznamné, lze počkat	Řešit co nejdříve
	Málo	Řešit co nejdříve	Řešit okamžitě
		Málo	Mnoho
		Kolik uživatelů postihlo	

Změny za provozu

- Mění se znalosti a dovednosti uživatelů. Ze začátečníků se stávají zkušení uživatelé.
- Postupný růst datové základny a často i počtu uživatelů mění i chování systému
- Je nutné sledovat chování systému za provozu
  - Analýza žurnálu (logy)
  - Vývoj názorů uživatelů (dotazníky, schůzky, interview)
  - Sledování reklamací
- Je nutné mít jasno, jak budeme na zjištěné problémy reagovat (budu peníze na úpravy, kdy je možné úpravy provést,..)
- Dotazníky – jednoduché, otázky uzavřené, vhodné vyplňovat na schůzkách

Podpora práce

- Poněvadž se uživatelé mění (přicházejí a odcházejí), a noví mohou mít i jiné zkušenosti je u větších systémů zajišťovat pravidelné školení uživatelů.
- Je třeba počítat se změnami hardwaru a SW a systém modernizovat i z tohoto důvodu

Výhody GUI

- Standard pro současné IS
- Možnost intuitivní práce, menší nároky na paměť, možnost učení metodou pokusů a omylů
- Pocit neustále interakce s počítačem
- Grafické rozhraní je nutností pro grafické aplikace. GUI je podmínkou pro vizuální metody programování.
- Znak modernosti systému

Nevýhody GUI

- Při rutinním používání relativně málo efektivní
- Neustálá interakce s počítačem, ergonomická náročnost
- Menší možnost tvorby skriptů, respektive opakovatelných sekvencí, a kontroly práce
  - Obrana: Tvorba maker, horké klávesy nebz znakové rozhraní u časo používaných činností

Životní cyklus rozhraní

- Poznání uživatele
- Analýza konkurenčních řešení
- Požadavky na uživatelnost (i s ohledem na finance)
- Návrh za účasti uživatelů (i noví)
- Realizace (vícenásobný návrh, syntéza, iterativní rozšiřování, zlepšování)
- Heuristická analýza
- Prototypování
- Testování
- Sledování za provozu