Základy tvorby interaktívnych aplikácií

Ing. Peter Drahoš, PhD.

LS 2013 - 2014

Obsah

- Priebežný test, správne odpovede
- HCI
  - Čo je to HCI, základné ciele
  - Odporúčania pre návrh rozhraní
  - Testovanie a evaluácia aplikácií
- Pauza: Nahliadnutie do testov
- Náhradný priebežný test

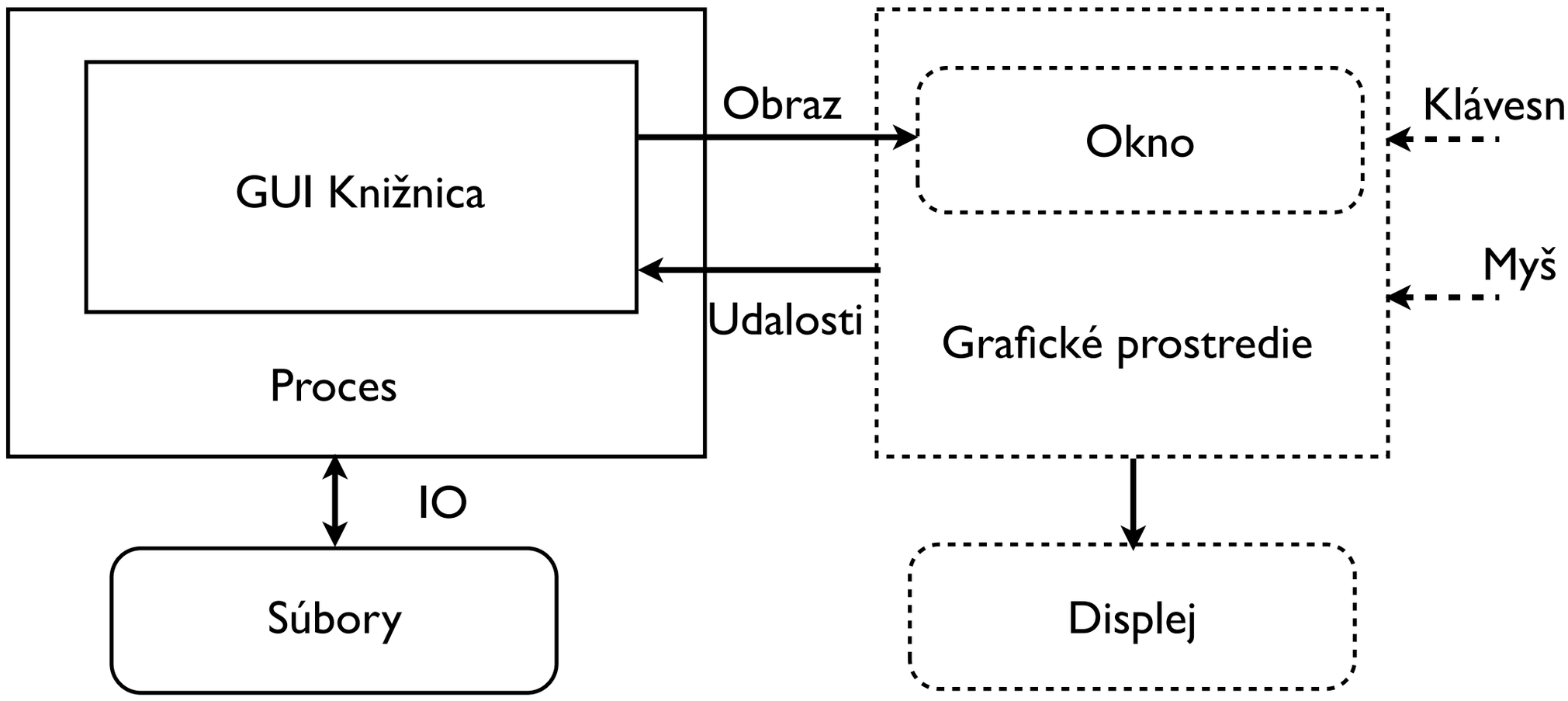
Priebežný test A

1. (2b) Nakreslite typickú štruktúru grafickej aplikácie s ohľadom na jej vstupy a výstupy.
2. (2b) Slovne opíšte čo je *MVC* a stručne vysvetlite jeho základné časti.
3. (2b) Vysvetlite čo je vzor *observer* a uveďte aspoň jedno jeho typické použitie v kontexte interaktívnych aplikácií.
4. (2b) Napíšte jednoduchý program v HTML5+JavaScript ktorý po načítaní dokumentu vypíše do *console* všetky parametre objektu *document*.
5. (2b) Definujte v jaz. JavaScript objekt *account* s parametrom *balance* a metódou *add* ktorá zvýši balance o hodnotu danú jej parametrom. Objekt musí obsahovať konštruktor ktorý nastaví počiatočnú hodnotu parametra balance.

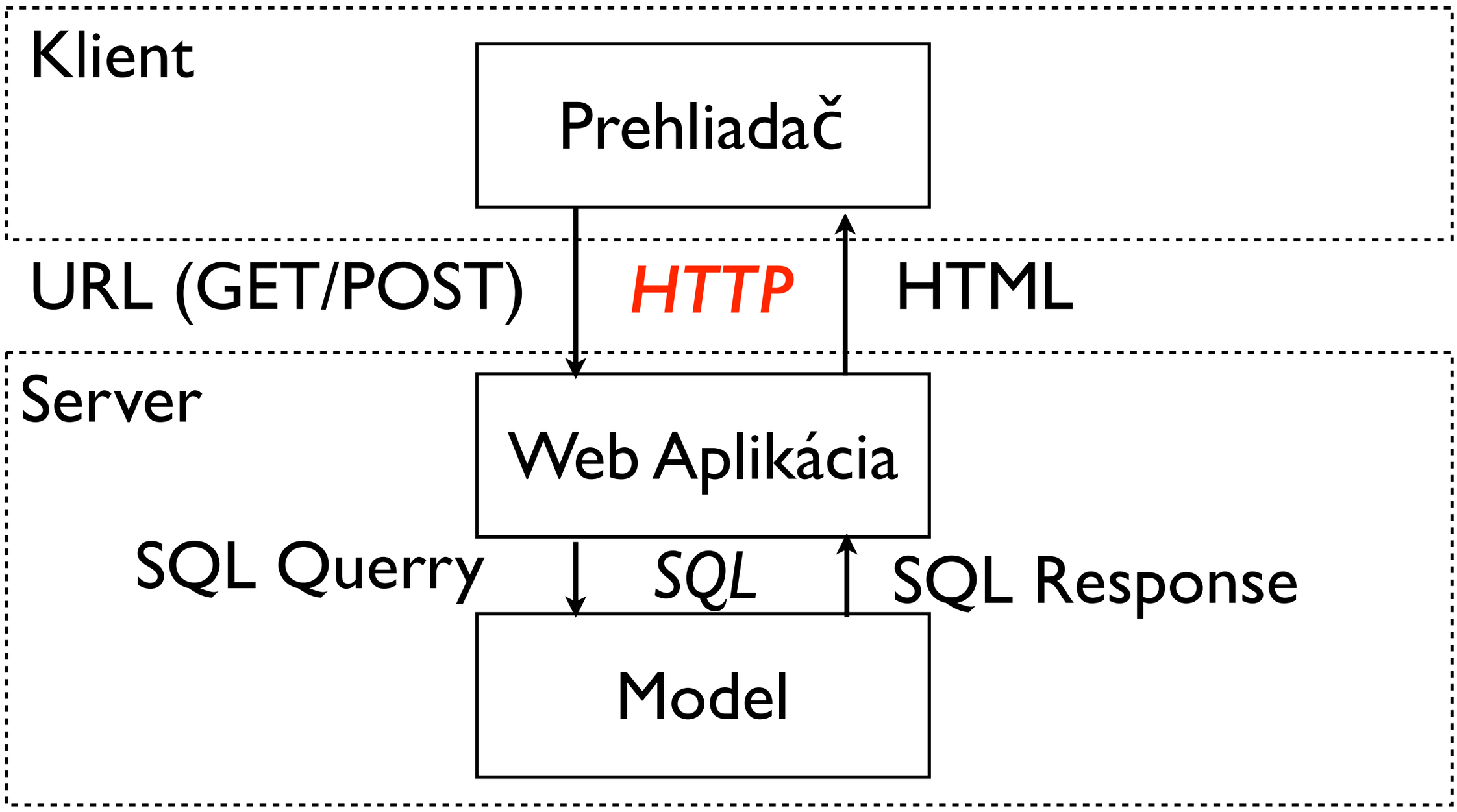
Priebežný test B

1. (2b) Nakreslite typickú štruktúru web aplikácie s ohľadom na jej vstupy a výstupy.
2. (2b) Slovne opíšte čo je to vzor *MVP* a stručne vysvetlite jeho základné vastnosti.
3. (2b) Vysvetlite čo je to *HTTP* a popíšte jeho formát pre dopyt typu *POST*.
4. (2b) Napíšte jednoduchý program v HTML5+JavaScript ktorý zmení textový obsah prvého paragrafu Red na “Green”.
5. (2b) Napíšte v jaz. JavaScript funkciu ktorá do konzoly vypíše *typ* prvého argumentu. Ak je argument typu *object* tak vypíše do konzoly i všetky jeho atribúty a ich hodnoty.

1. **A:** Nakreslite typickú štruktúru grafickej aplikácie s ohľadom na jej vstupy a výstupy.



1.B: Nakreslite typickú štruktúru web aplikácie s ohľadom na jej vstupy a výstupy.

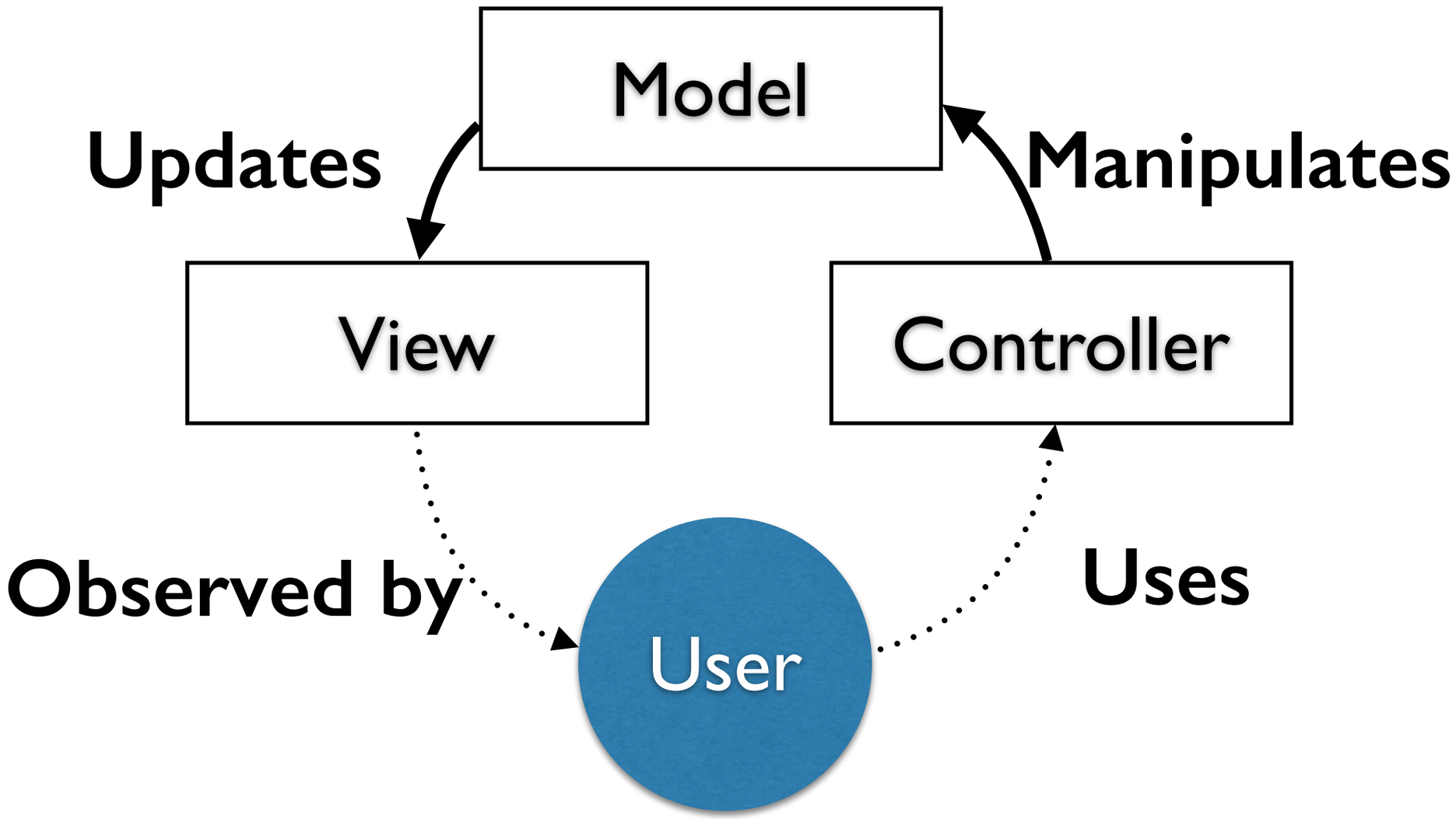


2. **A:** Slovne opíšte čo je *MVC* a stručne vysvetlite jeho základné časti.

- Architektonický návrhový vzor pre interaktívne aplikácie oddeľujúci dáta a logiku od ovládania a zobrazenia.

- *Model* - Aplikačná logika, dáta a hlavné algoritmy aplikácie
- *View* - Grafické či iné zobrazenie dát používateľovi
- *Controller* - Spracovanie ovládania aplikácie, ovláda model a view

MVC

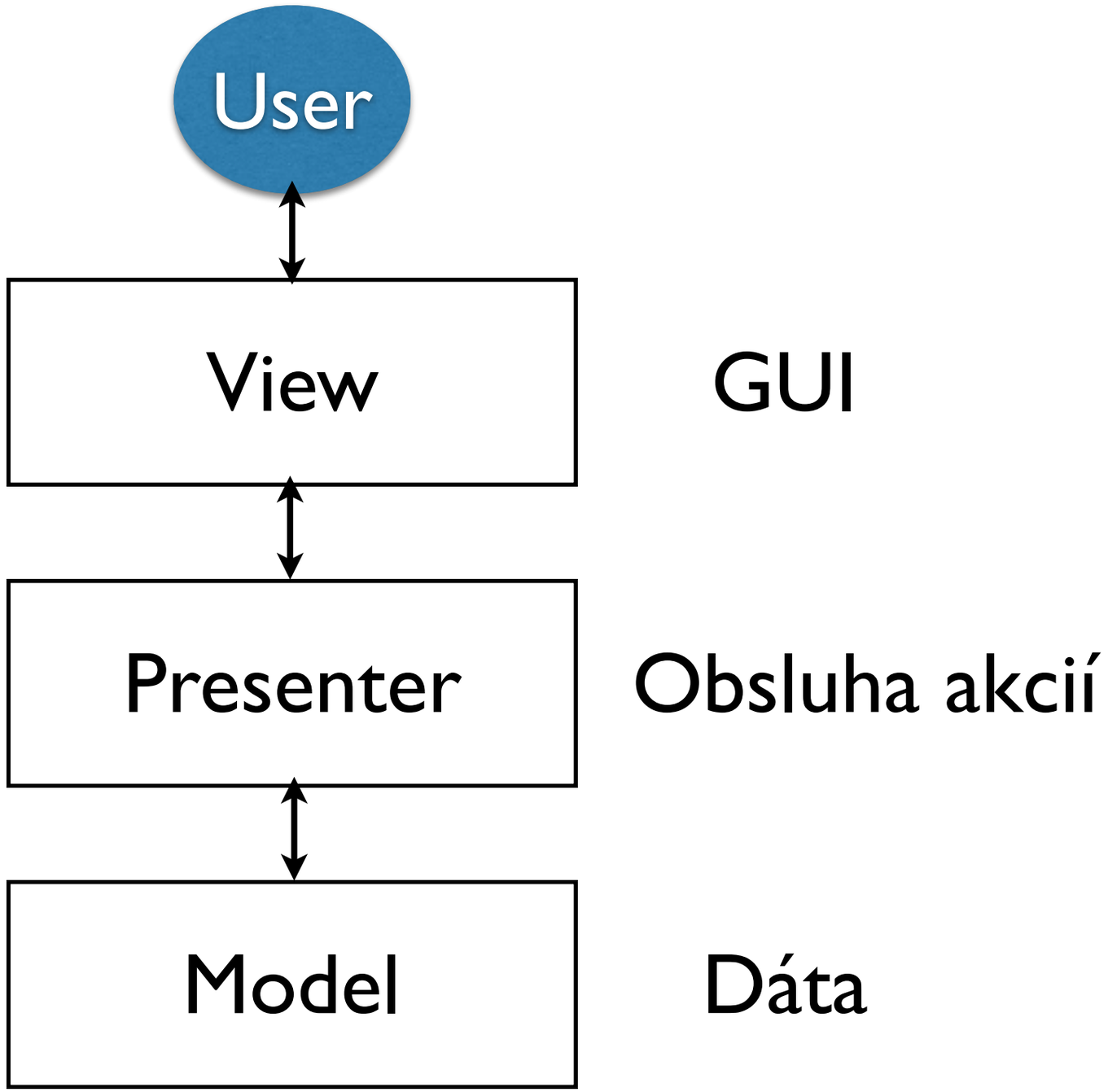

Model
Updates
Manipulates
View
Controller
Observed by
User
Uses

2.**B:** Slovne opíšte čo je to vzor *MVP* a stručne vysvetlite jeho základné vastnosti.

- Architektonický návrhový vzor pre interaktívne aplikácie oddeľujúci dáta a logiku od ovládania a zobrazenia. Podobný modelu MVC, trojvrstvová architektúra.

- *Model* - Aplikačná logika, dáta a hlavné algoritmy aplikácie
- *View* - Pevne dané grafické či iné zobrazenie dát používateľovi i s logikou interakcie
- *Presenter* - Obsluha akcií z *View* a príprava výstupov pre zobrazenie.

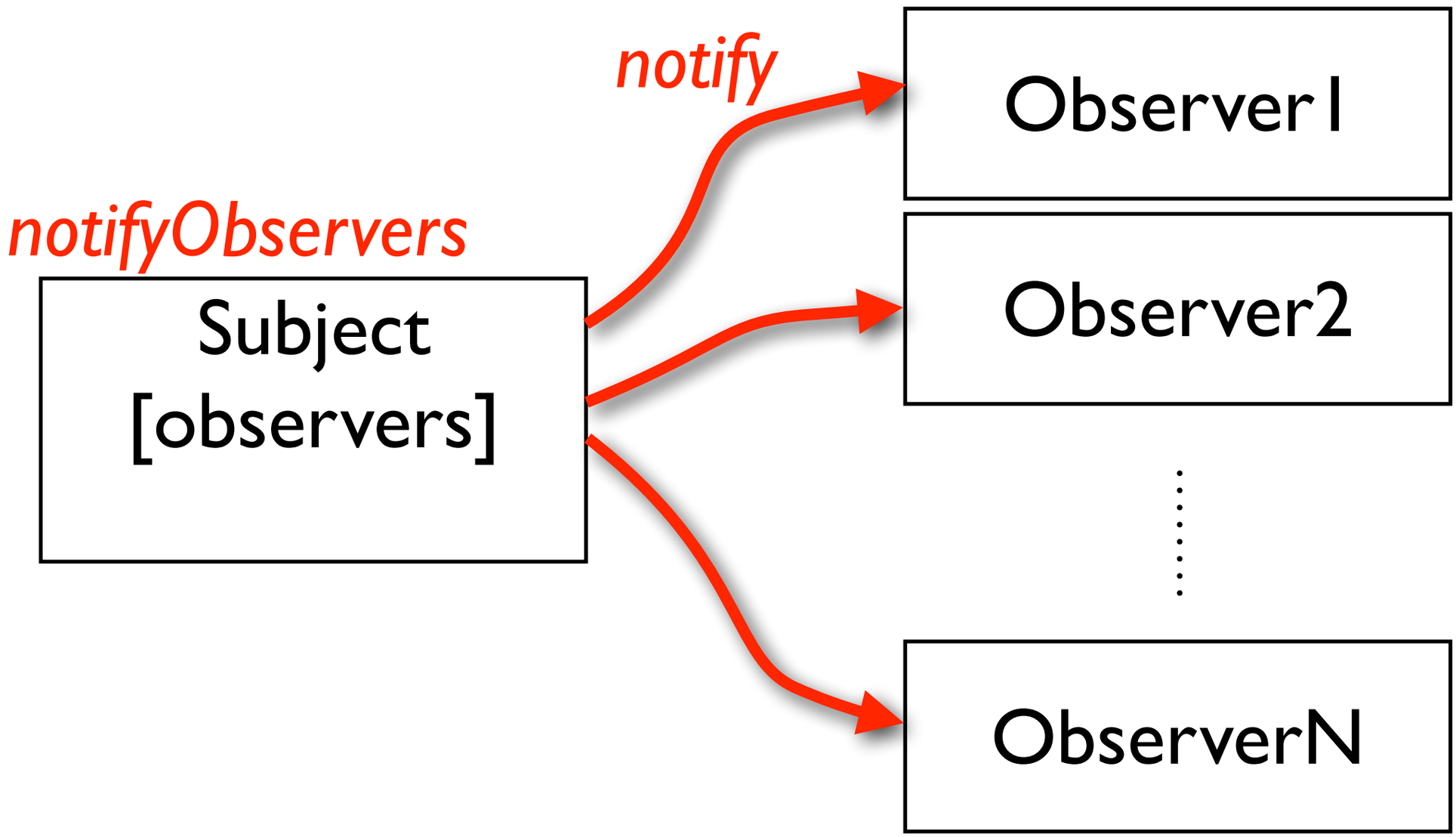
MVP



3. **A:** Vysvetlite čo je vzor *observer* a uveďte aspoň jedno jeho typické použitie v kontexte interaktívnych aplikácií.

- Vzor observer je návrhovým vzorom pre tvorbu softvéru ktorý definuje objekt *observer* ktorý je schopný reagovať na zmeny v objekte *subject* s využitím zoznamu pozorovateľov ktorým sú zasielané správy.
- Typické použitie je najme v kontexte GUI aplikácií kde sa vzor využíva na propagáciu správ a udalostí.

Vzor Observer



3. **B:** Vysvetlite čo je to *HTTP* a popíšte jeho formát pre dopyt typu *POST*.

- Hyper Text Transfer Protocol - Definuje pravidlá komunikácie prehliadača so serverom.
- Metóda POST - Určená na zaslanie informácie (en. Posting). Môže obsahovať veľké množstvo dát.
- Typické použitie - Formuláre, upload súborov

POST Formát

POST /index.html HTTP/1.1
HLAVICKA1: HODNOTA
HLAVICKA2: HODNOTA
...
HLAVICKAn: HODNOTA

TELO

4. **A:** Napíšte jednoduchý program v HTML5+JavaScript ktorý po načítaní dokumentu vypíše do *console* všetky parametre objektu *document*.

HTML5 + JavaScript

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>HTML5+JavaScript</title>
    <meta charset="UTF-8">
    <script>
    window.onload = function() {
      for (var i in document) { console.log(document[i]) }
    }
    </script>
  </head>
  <body>
  </body>
</html>
```

4. **B:** Napíšte jednoduchý program v HTML5+JavaScript ktorý zmení textový obsah prvého paragrafu Red na “Green”.

HTML5 + JavaScript

```
<html>
  <head>
    <title>HTML5+JavaScript</title>
    <meta charset="UTF-8">
    <script>
    window.onload = function() {
      var p = document.getElementsByTag("p")
      p[0].innerHTML = "Green"
    }
    </script>
  </head>
  <body>
   <p>Red</p>
  </body>
</html>
```

5. **A:** Definujte v jaz. JavaScript objekt *account* s parametrom *balance* a metódou *add* ktorá zvýši balance o hodnotu danú jej parametrom. Objekt musí obsahovať konštruktor ktorý nastaví počiatočnú hodnotu parametra balance.

JavaScript

```
function Account(balance) {
  this.balance = balance
  this.add = function(sum) {
    this.balance += sum
  }
}
```

5. **B:** Napíšte v jaz. JavaScript funkciu ktorá do konzoly vypíše *typ* prvého argumentu. Ak je argument typu *object* tak vypíše do konzoly i všetky jeho atribúty a ich hodnoty..

JavaScript

```
function vypis(obj) {
   var typ = typeof(obj)
   if (typ == "object") {
      for ( i in obj )
         console.log( i + ": " + obj[i] )
   } else {
      console.log(typ)
   }
}
```

HCI

- HCI - Human Computer Interaction
- Používatelské rozhranie nie je iba o usporiadaní výstupov/médií na obrazovke
- Cieľom je navrhnúť a ovládať všetky aspekty komuikácie. "*Look and Feel*"
- Psychológia: Tvorba mentálnych modelov aplikácie
- Ergonómia: Návrh ovládania

Prečo HCI?

- Používatel hodnotí aplikáciu podla jej výzoru a ovládania. Nie podla jej schopností
  - Používatelské rozhranie by malo zohladnovať schopnosti, skúsenosti a očakávania používatelov
- Zle navrhnuté rozhrania môžu spôsobit katastrofické zlyhania aplikácie.
  - Zle navrhnuté používatelské rozhranie je najčastejším dôvodom prečo sa aplikácie nepoužívajú.
- "*A user interface is well-designed when the program behaves exactly how the user thought it would*." – Joel Spolsky

Prečo HCI?

- Dobré používatelské rozhranie je neviditelné.
- Zlé rozhrania spôsobuju nespokojnosť používatelov.
- “*What was this product designer thinking?*”

Ciele HCI

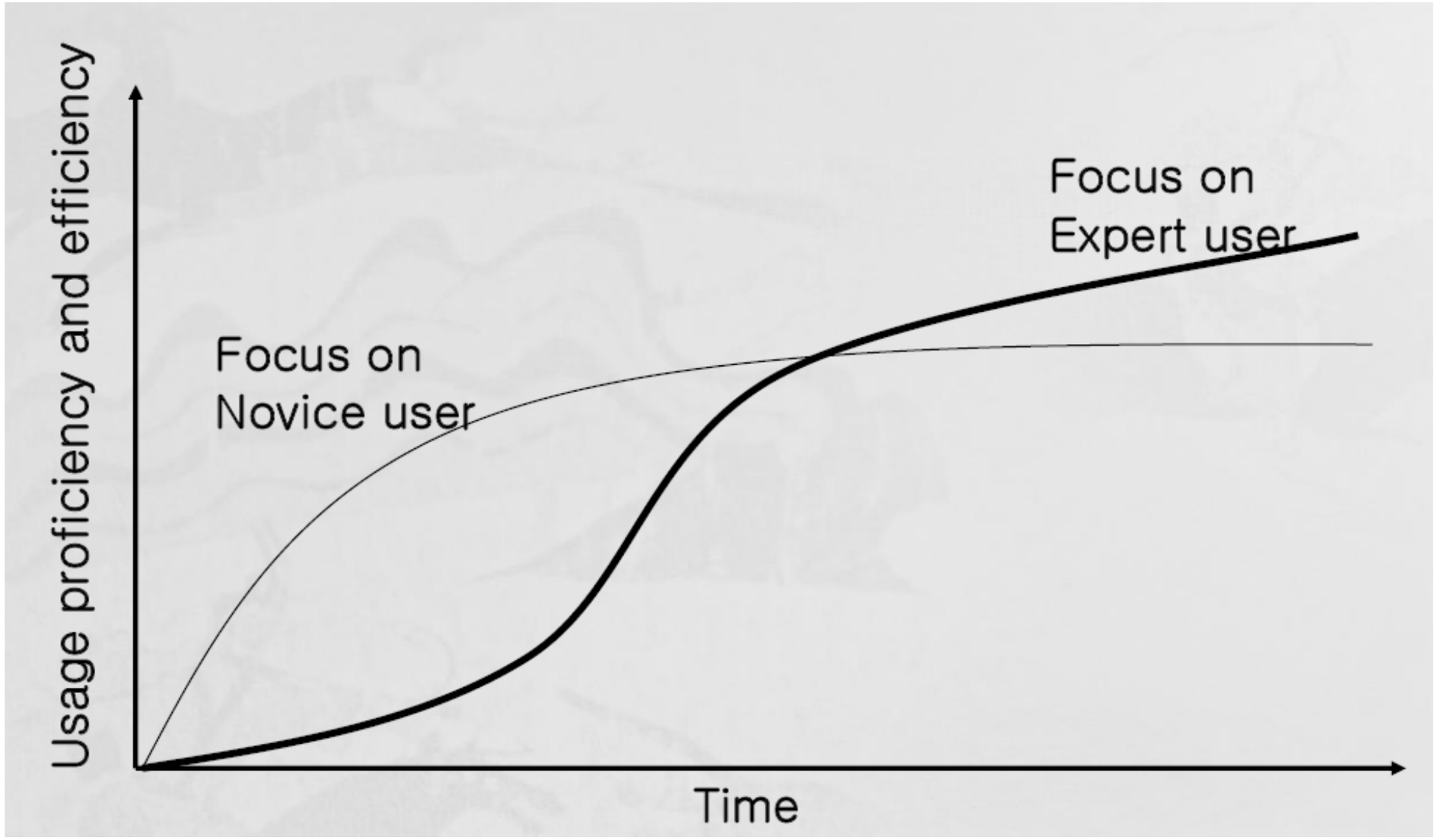
- Dobré používatelské rozhranie umožnuje jednoduché, prirodzené a pohodlné spracovanie úhohy pri komunikácii so systémom
- Umožnuje používatelom spracovať ich úholy efektívne
- Akým spôsobom však rozlýšime dobré od zlého?

Ciele HCI - Usability

- Usability - použitelnosť je jedno-dimenzionálna vlastnosť používatelského rozhrania
  - Learnability - Naučitelnosť, používaťel je schopný so systémom pracovať v krátkom čase
  - Efficiency - Efektivita, používatel je schopný so systémom pracovať s vysokou efektivitou
  - Memorability - Prerušený používatel je schopný pokračovať v úlohe bez nutnosti začať znova
  - Errors - Jasná indikácia chyby a jednoduchosť zotavenia sa aplikácie

Learning Curve

- Krivka naučitelnosti



Ciele HCI - Memory

- Schopnosť pamätať si a využiť zapametané znalosti vhodným spôsobom.
- Nepametáme si všetko, potrebujeme inforácie triediť a filtrovať
- Kontext je klúčový vo vzťahu k pameti
- Rozpoznávanie je oveľa lahšie ako spomínanie si
  - CLI vs GUI
- Lepšie si pametáme obraz ako slová
  - Ikony vs. Názvy

Ciele HCI - Human Factors

- Obmedzená krátkodobá pamet
  - Človek si pametá častokrát len 7 druhov informácie naraz. Ak aplikácia vyžaduje viac tak nastávajú chyby.
- Človek robí chyby
  - Každý človek občas urobí chybu. Neprimerané reakcie systému (alarm, záhadné správy atd.) vzyšujú stres používatela čo vedie k zvýšeniu pravdepodobnosti dalšej chyby
- Ľudia sú odlišný
  - Fyzické a psychické schopnosti používaťelov sa môžu výrazne líšiť. Tvorcovia rozhraní musia preto dbať aby nenavrhovali rozhrania s ohľadom len na ich skúsenosti a schopnosti.
- Rozdielne preferencie
  - Niektorý radi text, iný radšej GUI

Základy HCI

- User - System interaction, interakcia so systémom
  - Treba zohľadniť dve typické situácie
    - Akým spôsobom prenášať informáciu od používatela do systému
    - Akým spôsobom poskytnúť informáciu zo systému používatelovi
- Častokrát obe situácie riešime spoločne. Napríklad návrhom interakčnej metódy formou metafory.
  - Eg. Desktop Metafora

Interakčný styl

- Priama manipulácia
- Menu
- Formulár
- Príkazový riadok
- Prirodzený jazyk

Druh rozhrania

- Automatizované: bez interakcie
- Príkaz: jednorozmerné
- Grafická obrazovka: dvojrozmerné
- Graphics rozhranie: dva a pol rozmerné
- Budúcnosť: troj a viacrozmerné

Zlaté pravidlá

- Používatel musí mať vždy kontrolu
- Obmedz pameťové zataženie používatela
- Buď konzistentný

Používatel musí mať vždy kontrolu

- Navrhni interakciu tak aby nebolo nutné vykonávať nežiaduce alebo zbytočné akcie
- Umožni flexibilnú interakciu
- Interakcia my mala byť prerušiteľná a navrátitelná
- Prispôsob interakciu rozvoju schopnostiam používatela a umožni systém prispôsobovať
- Bežnému používatelovi nezobrazuj interné fungovanie apikácie
- Všetky zobrazené objekty by mali byť priamo interaktívne.

Obmedz pameťové zataženie

- Zníž zaťaženie krátkodobej pamete
- Používaj zmysluplné prednastavenie
- Definuj intuitívne skratky
- Rozhranie by malo sledovať metaforu z reálneho života
- Informácie poskytuj používatelovi postupne

Buď konzistentný

- Poskytuje zmysluplný kontext pre používatelovu úlohu
- Buď konzistentný hlavne v prípade viacerých aplikácií
- Predošlá skúsenosť používatelov formujuje ich očakávania, nemeň nič pokial to nie je nevyhnutné

Základných 10

- Kompaktný a prirodzený dialóg
  - Grafický dizajn a farba
  - *“Less is more”*
- Komunikuj v jazyku používatela
- Vyžaduj čo najmenej pamete používatela
- Consistencia
- Vždy poskytuj odozvu (*feedback*)

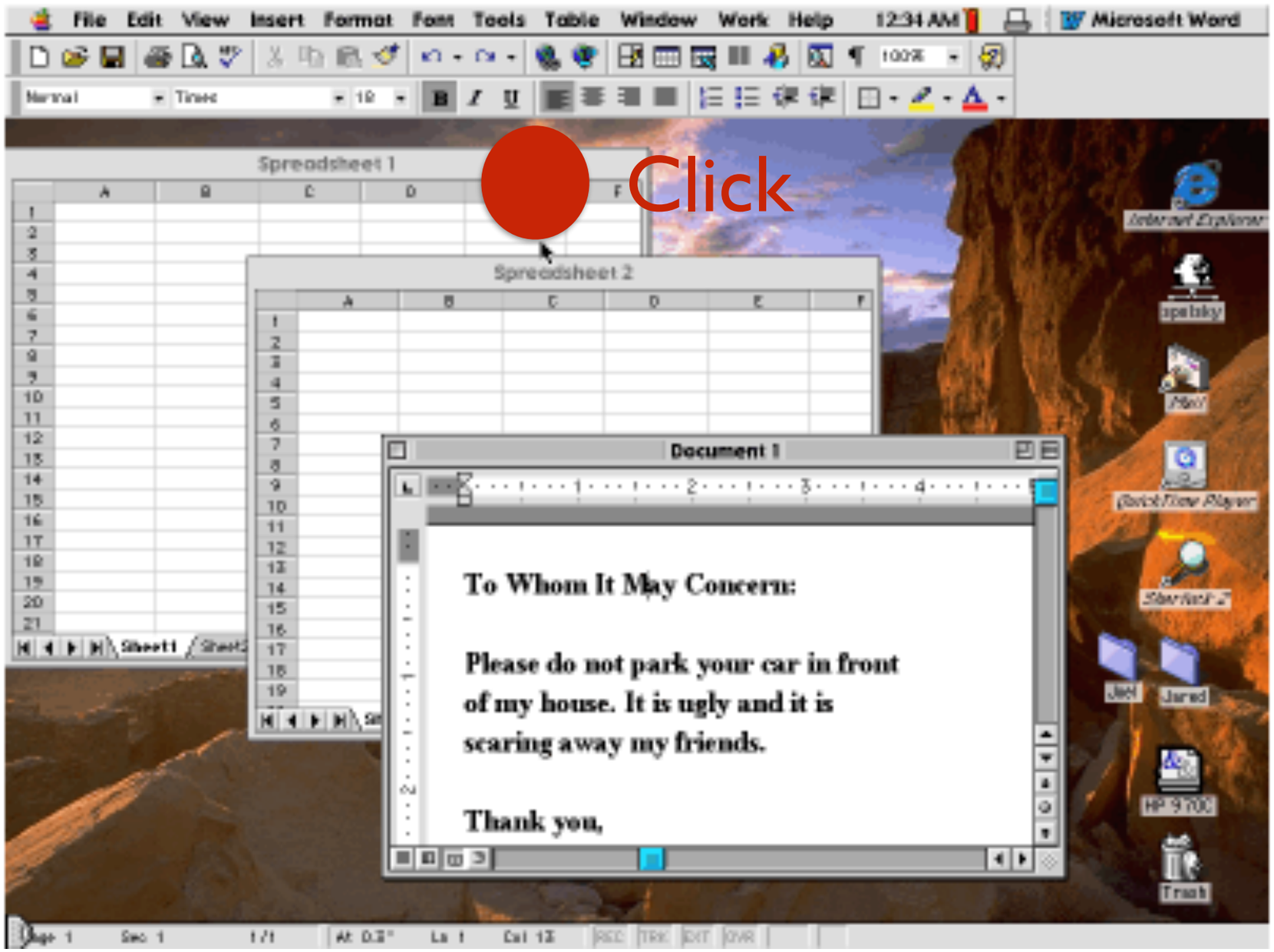
Základných 10

- Jasne vyznač a upozorni na koniec akcie a ukončenie
- Použi rozumné skratky
- Dobre formuluj chybové hlásenia
- Predchádzaj chybám
- Vždy poskytuj pomoc a dokumentáciu

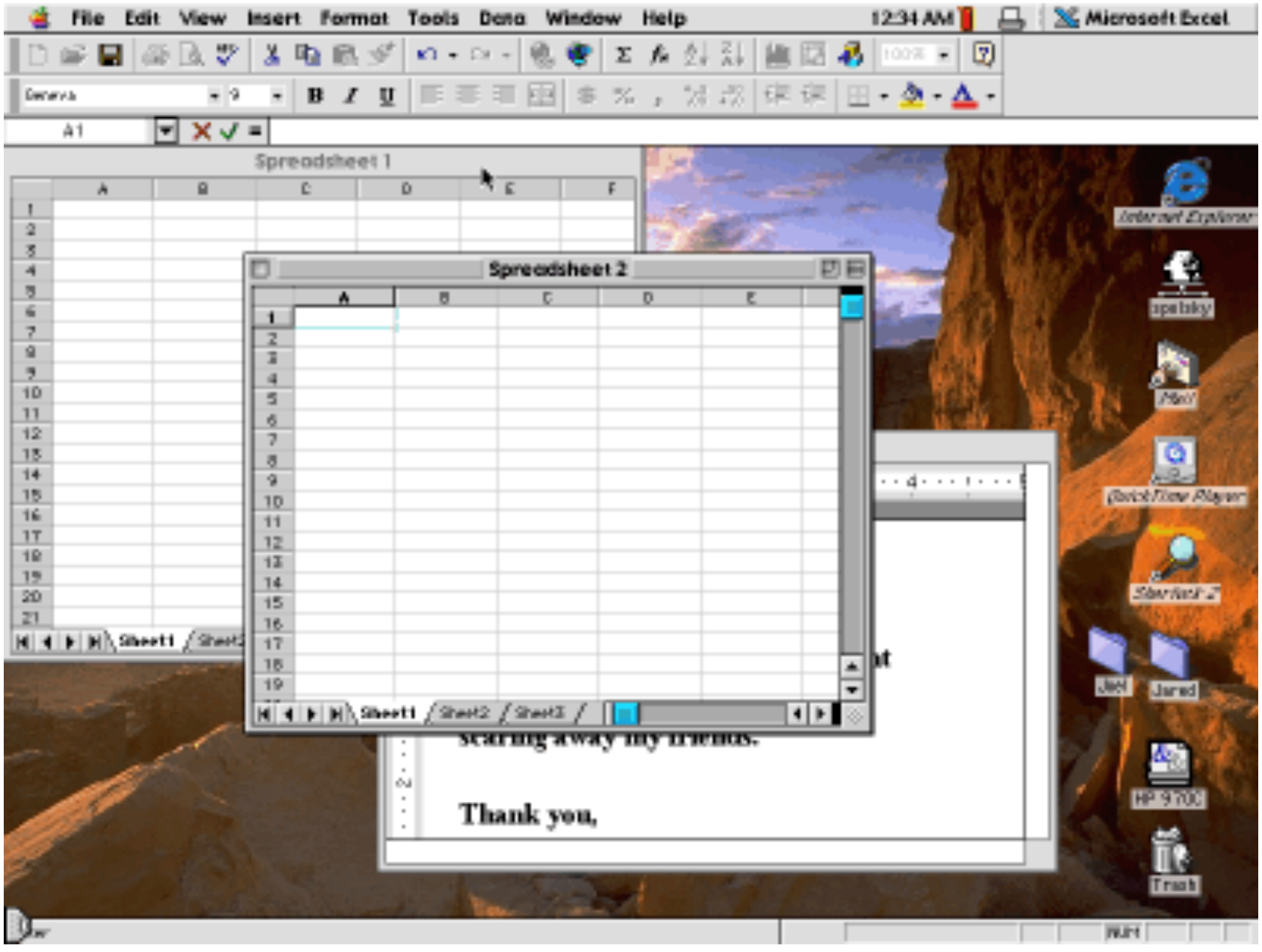
Zopár príkladov

- Nevyžiadaná pomoc

Zopár príkladov

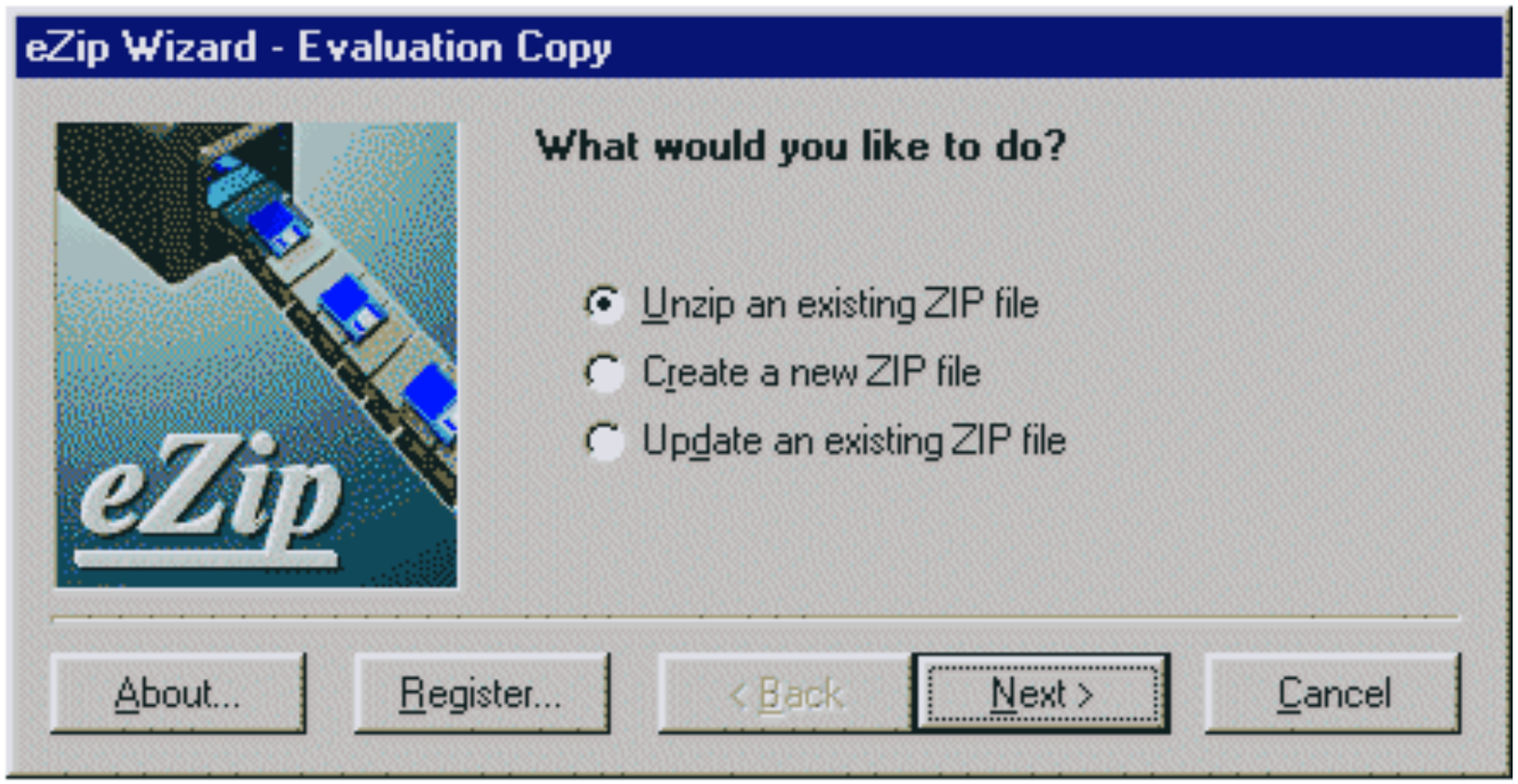


Zopár príkladov



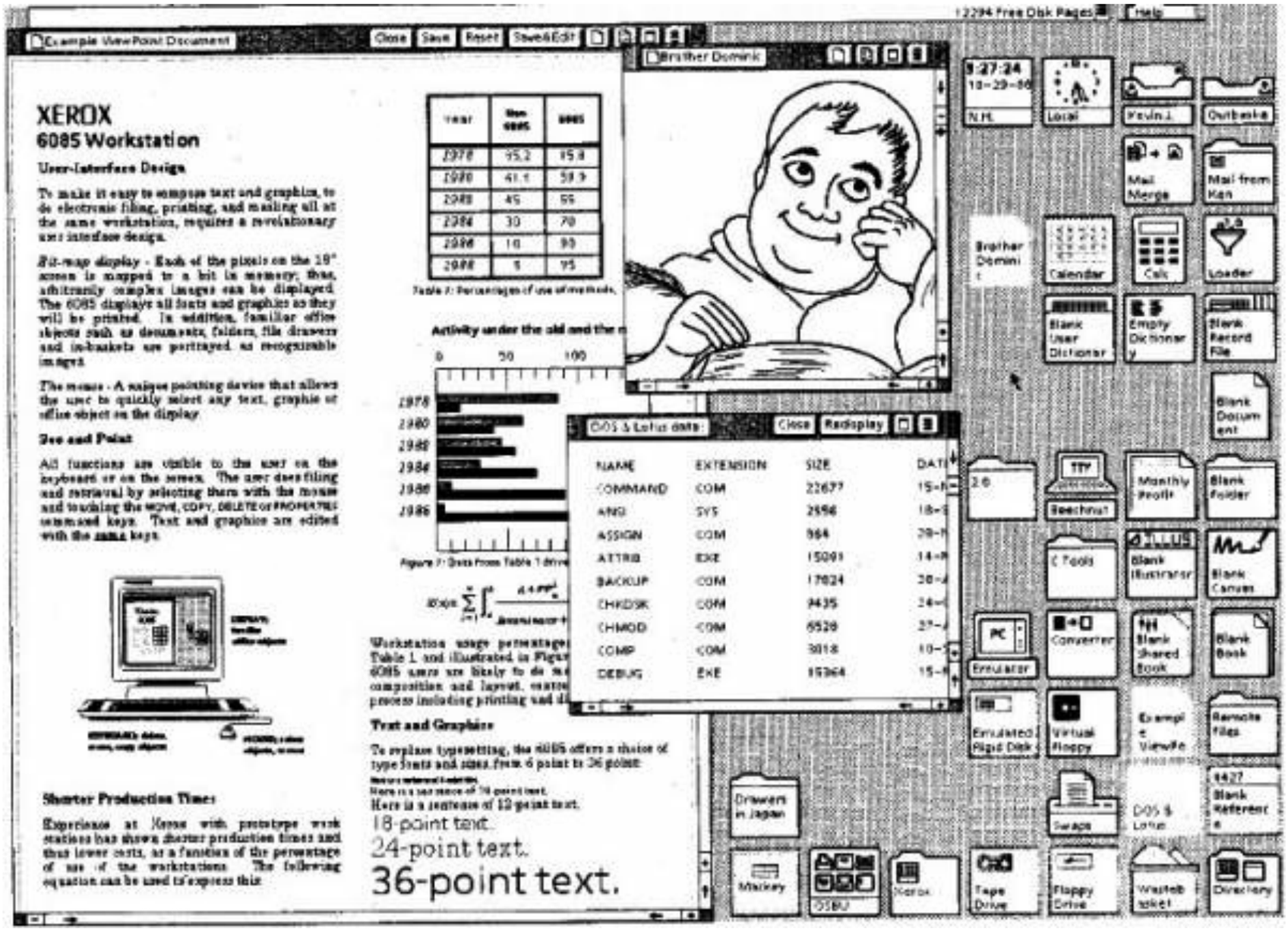
Zopár príkladov

- Zlý sprievodca



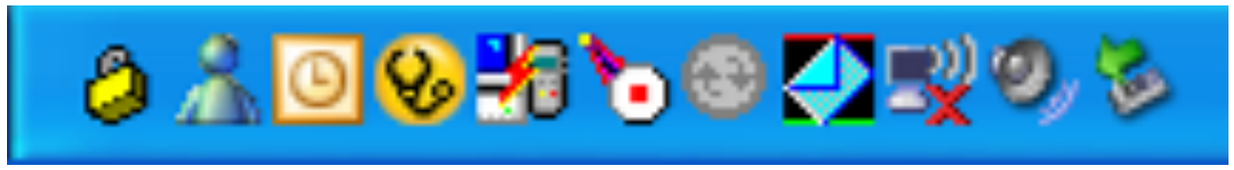
Zopár príkladov

- XEROX UI (1981)



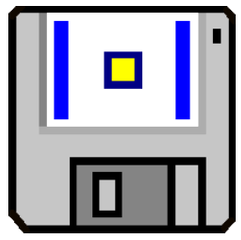
Zopár príkladov

- Koľko z nich poznáte?
- Aký stav indikujú?



Zopár príkladov

- Čo znamená daná ikona?
- Je to jasné každému používatelovi?

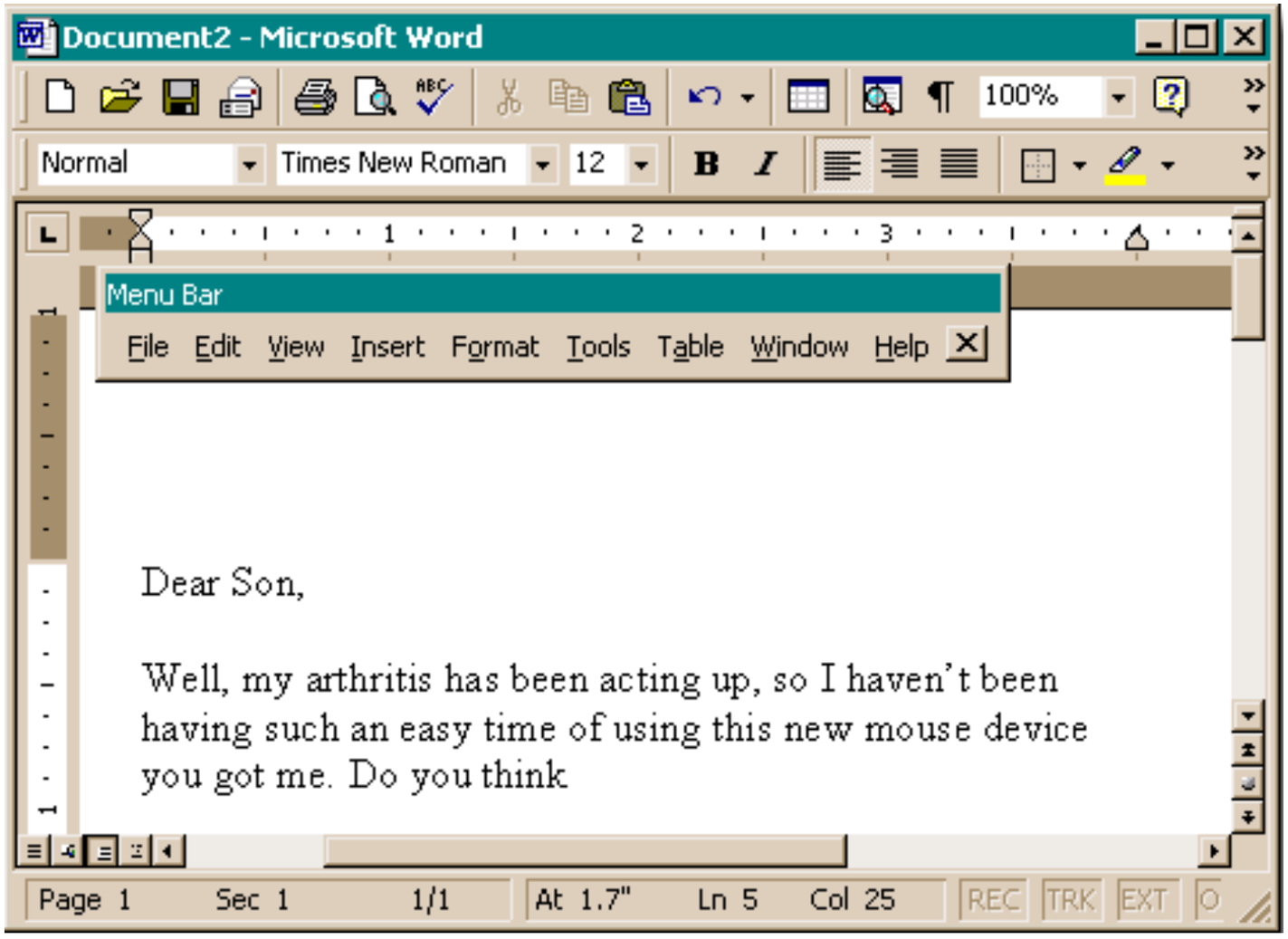


Zopár príkladov

- Konzistencia

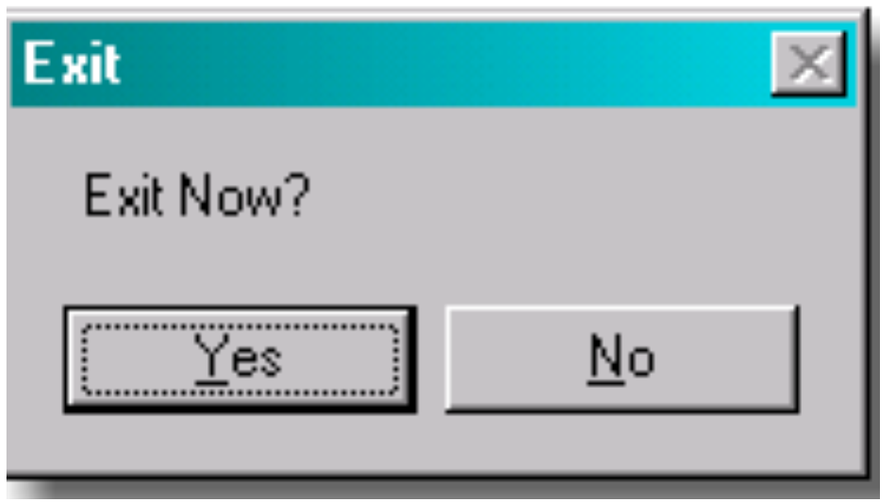
Zopár príkladov

- Zlá Implementácia
- Ako dať speť menu?



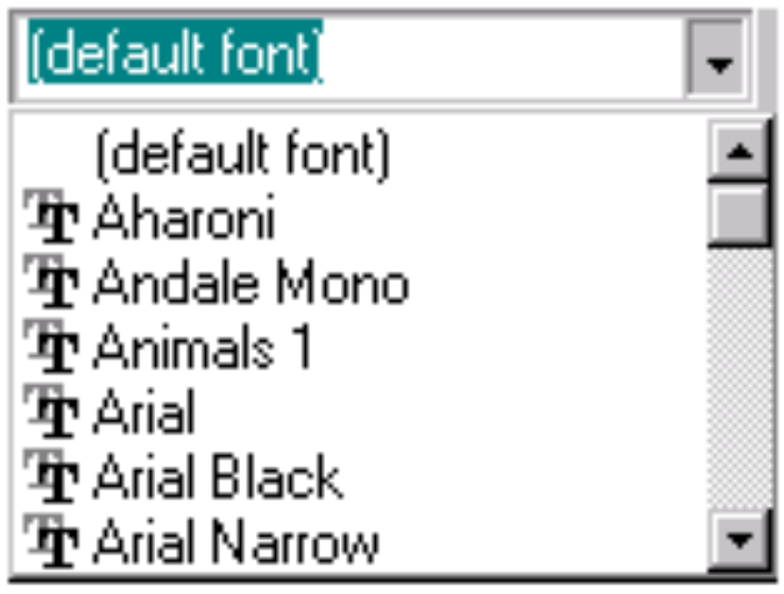
Zopár príkladov

- Farby a čitatelnosť
- Je to naozaj nutné potvrdiť?

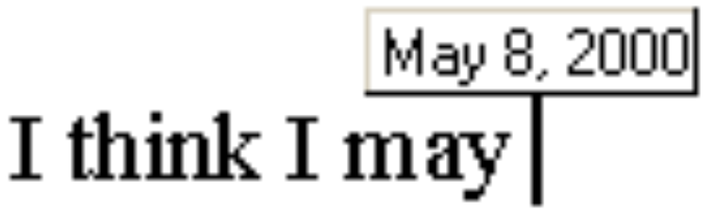


Zopár príkladov

- Rozumné prednastavenie
- Ok, ale aký to je font?



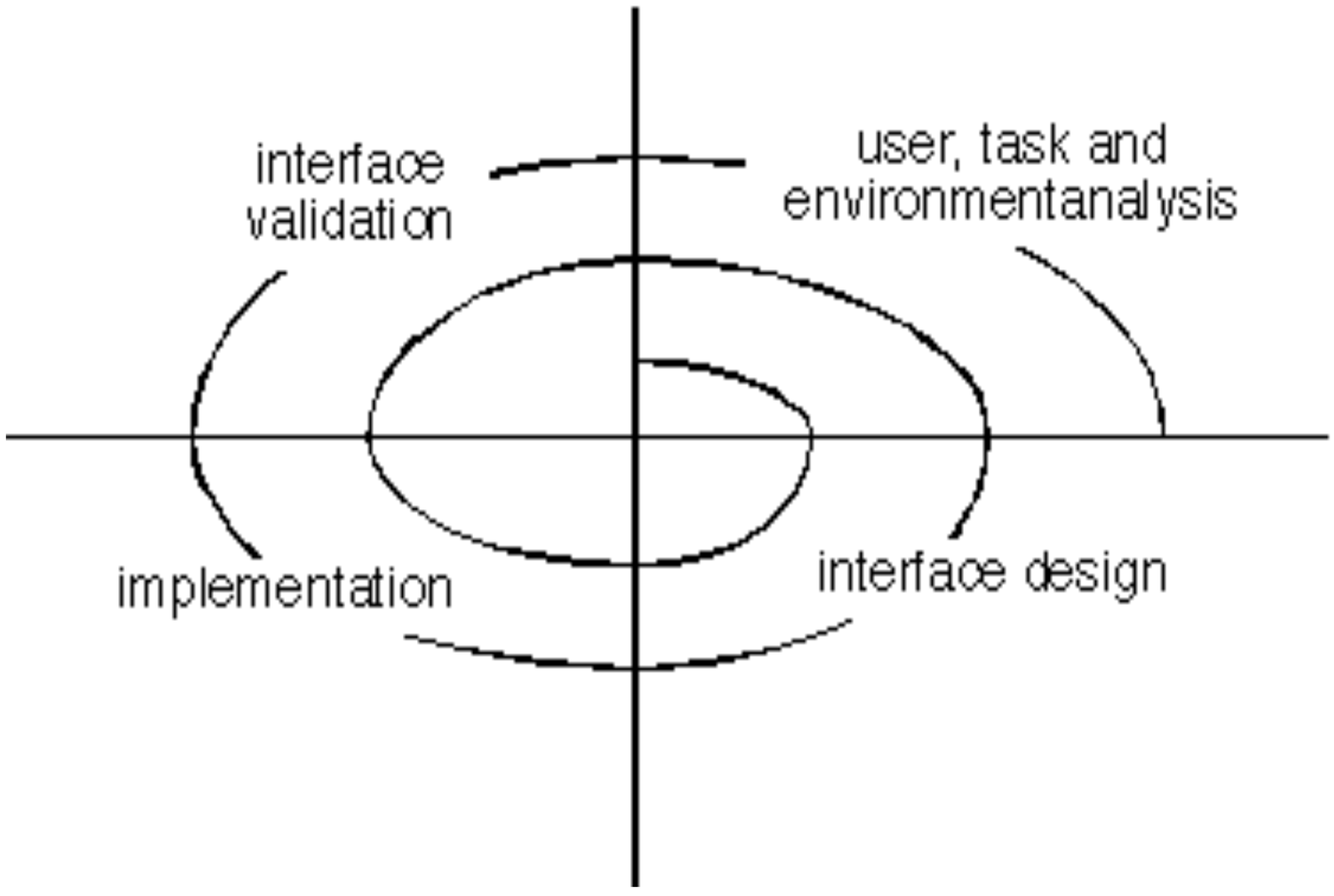
- Thank you, so helpful!



Návrh rozhrania

- 4 základné aktivity:
  - Identifikuj potreby používatelov a vytvor zoznam požiadaviek
  - Vytvor zopár alternatívnych návrhov
  - Vytvor interaktívny prototyp
  - Vyhodnoť dany návrh

Návrh rozhrania



Tvorba návrhu

- Použi zoznam požiadaviek od používaťelov a analyzuj predošlé rozhrania. Definuj objekty a akcie (operácie)
- Definuj udalosti (akcie používaťela) ktoré menia stav riešenia problematiky a rozhranie. Možné situácie over na modeli.
- Navrhni výzor rozhrania tak ako bude prezentované používatelovi
- Indikuj akým spôsobom očakávaš že systém bude reagovať na udalosti.

Vytvor prototyp

- High-fidelity: HTML, Flash, Rychly prototyp
- Low-fidelity: paprier, ceruzka, farby, lepidlo
- Výhody low-fidelity prototypu?
  - Lacné a rýchle zhotovenie
  - Veľmi lahké vykonanie zmien
- Nevýhody low-fidelity prototypu?
  - Nesimuluje reálne odozvy počítača

Vyhodnotenie

- HCI Lab