How to measure and to quantify usability of user interfaces
Rauterberg, G.W.M.

Published in:

Published: 01/01/1996

Document Version
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Please check the document version of this publication:
• A submitted manuscript is the author's version of the article upon submission and before peer-review. There can be important differences between the submitted version and the official published version of record. People interested in the research are advised to contact the author for the final version of the publication, or visit the DOI to the publisher's website.
• The final author version and the galley proof are versions of the publication after peer review.
• The final published version features the final layout of the paper including the volume, issue and page numbers.

Link to publication

Citation for published version (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

• Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
• You may not further distribute the material or use it for any profit-making activity or commercial gain
• You may freely distribute the URL identifying the publication in the public portal?

Take down policy
If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
HOW TO MEASURE AND TO QUANTIFY USABILITY OF USER INTERFACES

Matthias Rauterberg

Work and Organizational Psychology Unit (IfAP), Swiss Federal Institute of Technology (ETH)
Nelkenstrasse 11, CH-8092 Zürich, SWITZERLAND
Tel.: +41-1-6327082, Fax: +41-1-6321186, Email: rauterberg@ifap.bepr.ethz.ch

Keywords:
User-interface; analytical method; quantification; metrics

Abstract:
One of the main problems of standards in the context of usability of software quality is, that they can not be measured in product features. We present a new approach to measure user-interface quality in a quantitative way. First, we developed a concept to describe user-interfaces on a granularity level, that is detailed enough to preserve important interface characteristics, and is general enough to cover most of known interface types. We distinguish between different types of 'interaction-points'. With these kinds of interaction-points we can describe several types of interfaces (CUI: command, menu, form-fill-in; GUI: desktop, direct manipulation, multimedia, etc.). We carried out two different comparative usability studies to validate our quantitative measures. The results of one other published comparative usability study can be predicted. Results of six different interfaces are presented and discussed.

INTRODUCTION

We present a new approach to measure user-interface quality in a quantitative way. First, we developed a concept to describe user-interfaces on a granularity level, that is detailed enough to preserve important interface characteristics, and is general enough to cover most of known interface types (command language, CUI, GUI, multimedia, etc.). Different types of user-interfaces can be quantified and distinguished by the general concept of "interaction-points" (IPs). Regarding to the interactive semantic of IPs, different types of IPs must be discriminated.

An interactive system can be distinguished in a dialog and an application manager. So, we distinguish between dialog objects (DO, e.g. "window") and application objects (AO, e.g. "text document"), and dialog functions (DF, e.g. "open window") and application functions (AFIP, e.g. "insert section mark"). Each function \( f \in FS \), that changes the state of the content of an application object, is an application function. All other functions are dialog functions (e.g., window operations like move, resize, close). The complete set of all description terms is shown in defined in Table 1.

A dialog context (DC) is defined by all available objects and functions in the actual system state. If the set of available functions changes in the actual DC, then the system changes from one DC to another. In the actual DC all dialog objects (functions, resp.) are perceptible (PO, PF) or hidden (HO, HF). Four different mapping functions relate perceptible structures to hidden objects or functions (see Table 1).

Each interaction-point (IP) is related to at least one interactive function. If both mapping function's \( s \) and \( \alpha \) are of the type 1:m(\ any), then the user-interface is a command interface. If both mapping function's \( s \) and \( \alpha \) are of the type 1:1, then the user-interface is a menu or direct manipulative interface where each \( f \in FS \) is related to a perceptible structure PF (see Figure 1). The perceptual structure (visible, audible, or tactile) of a function (PF) can be, e.g., an icon, earcon, menu option, command prompt, or other mouse sensitive areas. The intersection of PF and PO is sometimes not empty: \( PF \cap PO \neq \emptyset \). Icons of graphical interfaces are elements of this intersection, e.g., PDFIP "copy" = PDO "clipboard", PAFIP "delete" = PAO "trash" (see Figure 1). Each interaction-point (IP) is related to at least one interactive function.
Table 1. The interaction space (IS) consists of the object (OS) and the function (FS) space

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>IS</td>
<td>interaction space</td>
</tr>
<tr>
<td>DC</td>
<td>dialog context</td>
</tr>
<tr>
<td>OS</td>
<td>object space</td>
</tr>
<tr>
<td>FS</td>
<td>function space</td>
</tr>
<tr>
<td>PO</td>
<td>(perceptible) representations of objects</td>
</tr>
<tr>
<td>HO</td>
<td>hidden objects</td>
</tr>
<tr>
<td>PF</td>
<td>(perceptible) representations of functions</td>
</tr>
<tr>
<td>HF</td>
<td>hidden functions</td>
</tr>
<tr>
<td>PDFIP</td>
<td>(perceptible) represented DFIP</td>
</tr>
<tr>
<td>PAFIP</td>
<td>(perceptible) represented AFIP</td>
</tr>
<tr>
<td>IP</td>
<td>interaction-points</td>
</tr>
<tr>
<td>PDO</td>
<td>[IPs of dialog functions]</td>
</tr>
<tr>
<td>PAO</td>
<td>[IPs of application functions]</td>
</tr>
</tbody>
</table>

\[ IS := OS \times FS \]
\[ DC \subseteq IS \]
\[ OS := PO \cup HO \]
\[ FS := PF \cup HF \]
\[ PO := PDO \cup PAO \]
\[ HO := HDO \cup HAO \]
\[ PF := PDFIP \cup PAFIP \]
\[ HF := HDFIP \cup HAFIP \]
\[ PDFIP := \{(df, pf) \in HDFIP \times PF: pf = \delta(df)\} \]
\[ PAFIP := \{(af, pf) \in HAFIP \times PF: pf = \alpha(af)\} \]
\[ IP := DFIP \cup AFIP \]
\[ AFIP := PAFIP \cup HAFIP \]
\[ \delta := \text{mapping function of a } df \in HDFIP \text{ to an appropriate } pf \in PF. \]
\[ \alpha := \text{mapping function of an } af \in HAFIP \text{ to an appropriate } pf \in PF. \]
\[ PDO := \{(do, po) \in HDO \times PO: po = \mu(do)\} \]
\[ PAO := \{(ao, po) \in HAO \times PO: po = \nu(ao)\} \]
\[ \mu := \text{mapping function of a dialog object } do \in DO \text{ to an appropriate } po \in PO. \]
\[ \nu := \text{mapping function of an application object } ao \in AO \text{ to an appropriate } po \in PO. \]

Figure 1. An actual dialog context (DC) of a direct manipulative interface with the representation space of the interactive object (PAO: e.g., data window; PDO: e.g., trash), and the representation space (PF: marked by circles) of the interactive functions (PAFIP: e.g., pop-up menu, trash; PDFIP: e.g., window scrolling).

FOUR QUANTITATIVE MEASURES OF INTERFACE ATTRIBUTES

One important difference between user-interfaces can be the "interactive directness". A user-interface is 100% interactively direct, if the user has fully access in the actual dialog context to all \( f \in FS \) (see Ulich et al., 1991). This is the case for all command language interfaces. Another important interface attribute is the amount of "feedback". Good interface design is characterised by optimising the multitude of DFIPs (e.g. "flatten" the menu tree, and by allocating an appropriate PDFIP to the remaining HDFIPs. One disadvantage of snapshots (cf. Figure 1) is that all hidden structures could not be referenced. To describe the hidden functionality a schematic view is needed (cf. Rauterberg, 1995).

To estimate the amount of "feedback" of an interface a ratio is calculated: "number of PFs" (\#PF = \#PDFIP + \#PAFIP) divided by the "number of HF's" (\#HF = \#HDFIP + \#HAFIP) per dialog context. This ratio quantifies the average "amount of functional feedback" of the function space (FB; see Formula 1). We abbreviate the number of all different dialog contexts with D. A GUI has often a very large number of DCs. To handle this problem we take only all task related DCs into account. Doing this, our measures will give us only a lower estimation for GUIs.

The average length of all possible sequences of interactive operations (PATH) from the top level dialog context (DC, e.g., 'start context') down to DCs with the desired HAFIP or HDFIP can be used as a possible quantitative measure of "interactive directness" (ID, see Formula 2). The measure ID delivers two indices: one for HAFIPs and one for HDFIPs. A PATH has no cycles and...
has not more than two additional dialog operations compared with the shortest sequence. An
interface with the maximum ID of 100% has only one DC with path lengths of one dialog step. We
abbreviate the number of all different dialog paths with P.

**Functional feedback:**

\[
FB = \frac{1}{D} \sum_{d=1}^{D} \left( \frac{\#PF_d}{\#HF_d} \right) \times 100\% 
\]

**Application flexibility:**

\[
DFA = \frac{1}{D} \sum_{d=1}^{D} \left( \#HAFIP_d \right) 
\]

\[
ID = \left( \frac{1}{P} \sum_{p=1}^{P} \log \left( \text{PATH}_p \right) \right)^{-1} \times 100\% 
\]

**Interactive directness:**

**Dialog flexibility:**

\[
DFD = \frac{1}{D} \sum_{d=1}^{D} \left( \#HDFIP_d \right) 
\]

To quantify the flexibility of the application manager we calculate the average number of
HAFIPs per dialog context (DFA; see Formula 3). To quantify the flexibility of the dialog manage-
er we calculate the average number of HDFIPs per dialog context (DFD; see Formula 4). A
modeless dialog state has maximal flexibility (e.g., "command" interfaces). To interpret the re-
sults of our measures appropriately, empirical studies are necessary.

**RESULTS OF APPLYING THE MEASURES**

We carried out two different comparative usability studies to validate our measures (experiment-
I see Brunner, 1993, and experiment-II see Rauterberg, 1992). A third external comparative
study (experiment-III; cf. Grützmacher, 1987) was used for a cross-validation (for a more de-
tailed description see Rauterberg, 1995). All three investigated software products have the same
application manager, but two different dialog managers each.

Interesting is the fact, that the GUI of experiment-I supports the user with less "functional feed-
back" (FB = 66%, see Table 2) on average than the CUI (FB = 73%). This amount of FB of the
CUI is caused by 22 small DCs with FB = 100%; the GUI has only 14 DCs with FB = 100%.
The amount of functional feedback seems not to be related to the advantage of GUIs. There must
be another reason.

**Table 2.** Comparison our three empirical validation studies relating to the quantitative measures
ID, FB, DFA, and DFD. P is the number of all different dialog PATHs for an AFIP or a DFIP; D
is the number of all different DCs. ["I 1  » I 2": interface-1 is better than interface-2]

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Interface type and dialog structure</th>
<th>D</th>
<th>FB</th>
<th>ID(AFIP) %</th>
<th>ID(DFIP) %</th>
<th>DFA</th>
<th>DFD</th>
<th>empirical result (p-significance)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>CUI-hierarchical</td>
<td>36</td>
<td>73</td>
<td>24.7</td>
<td>23.2</td>
<td>12.1</td>
<td>10.1</td>
<td>CUI « GUI p ≤ .001</td>
</tr>
<tr>
<td>I</td>
<td>GUI-hierarchical</td>
<td>28</td>
<td>66</td>
<td>22.5</td>
<td>25.5</td>
<td>19.5</td>
<td>20.4</td>
<td></td>
</tr>
<tr>
<td>II</td>
<td>Multimedia-hierarchical</td>
<td>68</td>
<td>100</td>
<td>25.1</td>
<td>28.1</td>
<td>3.6</td>
<td>0.5</td>
<td>MM hier  = MM net p ≤ .085</td>
</tr>
<tr>
<td>II</td>
<td>Multimedia-net shaped</td>
<td>65</td>
<td>100</td>
<td>40.7</td>
<td>46.3</td>
<td>4.2</td>
<td>1.3</td>
<td></td>
</tr>
<tr>
<td>III</td>
<td>CUI-hierarchical</td>
<td>363</td>
<td>86</td>
<td>20.9</td>
<td>23.9</td>
<td>2.0</td>
<td>1.9</td>
<td>CUI hier  = CUI net p ≤ .825</td>
</tr>
<tr>
<td>III</td>
<td>CUI-net shaped</td>
<td>389</td>
<td>90</td>
<td>15.8</td>
<td>21.9</td>
<td>1.3</td>
<td>2.7</td>
<td></td>
</tr>
</tbody>
</table>

The "interactive directness" is not quite different between both interfaces: CUI: ID = 24.7% for
AFIPs and 23.2% for DFIPs versus GUI: ID = 22.5% for AFIPs and 25.5% for DFIPs (see Table
2). Only the two measures of "flexibility" show an important difference: CUI: DFA = 12.1 and
DFD = 10.1 versus GUI: DFA = 19.5 and DFD = 20.4 (see Table 2).

In the hierarchical dialog structure (MM_hier) of the multimedia information system (experiment-
II) only one way is given to reach an AFIP. In the net-shaped version (MM_net) several ways are
possible to navigate through the dialog structure. But, what is an AFIP in the context of a multimedia system? We define the application kernel of a multimedia system as the set of all masks with a relevant information in the sense of the main purpose of the information system (e.g., concrete information's about bank services in the context of a bank information system); all other masks are part of the dialog manager. A PAFIP is therefore each mouse sensitive area that changes the system to a mask of the application kernel; all mouse sensitive areas are DFIP's.

With the support of Grützmacher we were able to analyse all 752 dialog contexts for both interfaces of the simulation tool 'Moro' (cf. experiment-III). For the hierarchical CUI, we got the following results: DFA = 2.0 and DFD = 1.9; for the net-shaped CUNet: DFA = 1.3 and DFD = 2.7 (see Table 2). These results for DFA and DFD of both CUI interfaces give us a strong empirical evidence that the following assumptions are correct: (1) The dialog flexibility can be quantitatively measured in a task independent way, and (2) the values of DFA and DFD must exceed the threshold of 15.

DISCUSSION AND CONCLUSION

If our interpretation of the outcome of experiment-I is correct then we can not find a significant performance difference for dialog structures that remain under the assumed threshold of 15. To control the factor of feedback we carried out the second experiment with a multimedia information system that has 100% functional feedback for both interfaces (Brunner 1993). We picked out a multimedia information system with a hierarchical dialog structure where DFA and DFD are clearly under 15. We implemented a comparable system with a net-shaped dialog structure where DFA and DFD have nearly the same ratio of flexibility as in experiment-I: DFA_CUI / DFA_CUI = 1.6 and DFA_MMnet / DFA_MMhier = 1.2; DFD_CUI / DFD_CUI = 2.0 and DFD_MMnet / DFD_MMhier = 2.6. As we predicted, we could not find a significant performance difference between both types of dialog structures.

To make sure that our results are not biased by our own expectations, we carried out a cross validation study. To do this, (1) we need the outcomes of an external independent comparison study between two different interfaces and (2) the possibility to apply our quantitative measures to all DCs of both interfaces. The empirical investigation of Grützmacher (1987) fulfils both conditions (experiment-III). Given our interpretation of the experiment-I and -II we expected and found a value for DFA and DFD under 15 for experiment-III. We interpret the negative result of experiment-III to the effect that flexibility must exceed our threshold to be effective.

The presented approach to quantify usability attributes and the interactive quality of user-interfaces is a first step in the right direction. The next step is a more detailed analysis of the relevant characteristics and validation of these characteristics in further empirical investigations. In the context of standardisation we can use our criteria to test user-interfaces for conformity with standards.

REFERENCES


