Feature Checklists in HCI: some basic results

Edward A. Edgerton, Stephen W. Draper, Stephen B. Barton
GIST (Glasgow Interactive Systems cenTre)
University of Glasgow, Glasgow, U.K.
steve@psy.glasgow.ac.uk

Abstract
Feature checklists are a method of measuring the usage of commands by exploiting users' memories. The perceived usefulness of commands can also be measured, as can awareness of their existence and functions. Experiments found that their accuracy (validity) was greater than 80% in all cases. Increased visual realism of the presentation may increase this still further. Extensions to bugs and to task descriptions are discussed.

Introduction
There are at least 3 different meanings for "checklist" even in the human factors area. A checklist in itself is just a list of items that may be checked (ticked) off.

1. **Preflight checklist** To support behaviour by being an external memory aid e.g. aircraft pilots use one for preflight checks, to supplement recall of a large set or list (sequence) of items. The items are familiar and are instantly and reliably recognised; the checklist is to ensure none are forgotten. Each item is a cue for recall of an action to take.
2. **Usability checklist** To elicit information, not from the respondent's memory but via their behaviour i.e. a checklist of information-getting behaviour, the results of which are recorded. E.g. Ravden & Johnson's (1989) checklist for usability evaluation is intended for use by those not familiar with it: each item is not recognised but contains a full description of what to look for.
3. **Feature checklist** To elicit information from the respondent's memory: the checklist lists features (e.g. commands) of the machine (unlike (2)), and gathers information from a human memory.

This paper concerns feature checklists (FCs). They consist of a list of commands or other features of a user interface, which it is hoped the subject will immediately recognise and remember, used as a cue for asking questions about them. Thus the usual layout is a long list of features, against which are a few columns each asking a question about each feature. In that sense, they are a specialised questionnaire. The questions asked might include whether the command is used, how frequently, whether it is useful. This depends on people having no trouble remembering the commands in an interface, an assumption we set out to test.

The main application of FCs is to measure which commands are used, and how frequently. They are thus an alternative to command logging (journalling) systems built into the software. Ideally perhaps they would be unnecessary since it might be argued that all interface software should have such logging built in. In reality however, we are often asked to evaluate software which does not have such logging available, or for which it was not turned on in the period prior to the study. FCs use people's memories, which do not suffer from such drawbacks. The question is: are they sufficiently accurate?

In fact logging methods also have intrinsic weaknesses, and should in principle also be tested for accuracy. They record user actions, not intentions, and therefore record commands actually invoked, including those accidentally and even unknowingly invoked (e.g. by the mouse slipping on a pulldown menu). Thus they cannot be wholly trusted for calculations of which commands are known by a user, or which are useful. Some logging systems have other problems. Some record at a low level (e.g. mouse positions) and so create huge log files (and a translation problem) for a small amount of information at the command level. Others, such as Unix' **acct**, record processes which have a variable and indirect relationship to user commands.

Finally feature checklists can be used to ask other per-command questions such as whether the command is useful (or vital, or of incidental convenience), whether users know about it (but just hasn't used it because it is useless, or on the contrary would use it if only they had known about it). Thus FCs can not only provide usage frequency data, which is useful to designers and evaluators in a number of ways, but also data on knowledge and usefulness of commands, and so can address several design problems such as feature overload (a high ratio of commands provided to commands used and found useful), information delivery problems (commands judged useful but not known about) and reminding problems (commands said to be useful, known about when queried, yet not used).

Basic results
Four studies were done, with the basic aim of measuring the validity of feature checklists for HCI. The ultimate goal is to measure the command use and knowledge of experienced users. However the only unquestionable check on validity is behavioural: observing users using commands. Thus the first study compared direct observation with both FCs and software logging, and had therefore to use new users. It also compared the validity of checklists with that of open-ended questionnaires in order to replicate in the HCI domain the only previous study we had found (Belson & Duncan 1962). It tried the two instruments in both orders (on two groups of subjects), so order effects could be observed while also being

able to compare first instrument accuracies across subjects. It studied new users of the Word editor. The FC, when issued first, achieved 85% accuracy, while the open-ended questionnaire achieved 48% at best (and that was only when issued after the FC).

In a second and third study (mainly aimed at exploring the effect of visual realism — see below) accuracy levels were 80%- 84%, and 89-95%.

A fourth study used experienced users (though not on an HCI task but on usage of Glasgow's underground railway). It therefore could not do direct behavioural validity checks of remembered use. It was mainly designed to extend validity checks to other columns for feature checklists, however distractor items were used as a validity check for memory of the existence of features, while the other measures were validated against interviews. We found 87% accuracy for existence of features, and 84% for the question of whether the subject knew what a feature was for.

In summary, we found FCs to have at least an 80% accuracy across these four studies, and often higher.

Visual realism

The most obvious reason why FCs are so successful is that they require only recognition, while the questionnaires required recall. A further reason may be that subjects were performing a similar task to that encountered in the real life situation, i.e. recognising the command in the context of the same list of possible alternatives. Could accuracy be further improved by increasing the degree of visual realism of the cue? In our standard FCs, commands were simply referred to by their name: the same name as appeared on the actual menus, but not in the same typeface and without the same spatial layout and other screen cues. Our second and third studies compared our standard FC layout with printed screen dumps and with allowing users to refer to the actual application on screen. Perhaps because the standard layout is already near to the ceiling of performance, we achieved a statistically significant improvement only after the data from an outlier subject had been excluded. Our present belief is that it is possible to obtain a small increase in accuracy by increasing the visual realism of checklist format.

Further variations

We are in the process of extending this technique from commands to bugs in the interface. While thinkaloud protocols are effective for discovering the existence of bugs, design teams must often limit their efforts at remediation to the most serious problems. To make these decisions, data on frequency and severity are necessary. Provided bugs can be described on a checklist in a readily recognisable way, checklists might be used to gather this data.

"Semantic" checklists

A variation we have already tested is to present as a cue, not the name and appearance of a command, but a description or paraphrase of its function or effect. This reduces to zero the visual realism of the prompt, but might be expected to match more closely the conditions of actual use. This is because when using a command, a user presumably starts from an internal, mental description of the effect (or task, or goal) they wish to accomplish and from that retrieve (in conjunction with screen prompts) the display "name" of the command i.e. retrieve and use a task—>action mapping. This form of checklist thus abandons visual realism, but tries to match more closely the normal task conditions of command use. These checklists might be called "semantic" since they attempt to remind users of their own internal meanings for commands. Their potential drawback is whether users can recognise the paraphrases at all — these require comprehension of a new description which recognition of display names does not.

We ran a study comparing (among other conditions) "semantic" with our normal feature checklists, to see whether they would achieve a comparable accuracy. For many commands they did, but the pattern of the results suggests that this is actually the wrong question to ask. There is evidence that where subjects answered the "semantic" checklist "wrongly" they were actually reporting that they did know how to achieve that effect, though not by the direct command in fact present in the interface. In other words, these "semantic" checklists seem to be a powerful instrument for detecting functions which the user needs and achieves, but for which they have not discovered the optimal command. Further development and laboratory work is needed, but it now looks as if the best instruments may use both visual and "semantic" cues, and that comparison of answers will reveal not only what is used but what would be used if information delivery were improved in that interface.

Acknowledgements
The work summarised here was supported by a grant "Measurement of user interface performance" from IED (Information Engineering Directorate) and SERC (Science and Engineering Research Council) ref.: GR/F 39171 / IED4/1/1109. Our colleagues on the project, particularly Pat Jordan and Paddy O'Donnell, have been closely involved in this work, and are partly responsible for any merit it may contain.

References

Belson, W. & Duncan, J.A. (1962) "A comparison of the check-list and the open response questioning systems" Applied statistics vol.11 pp.120-132

Ravden, S. & Johnson, G. (1989) Evaluating usability of human-computer interfaces: a practical method (Ellis Horwood: Chichester)