Interface design in computer-based language testing

Glenn Fulcher *University of Dundee*

There is no published material in the language testing literature on the process of, or good practice in, developing an interface for a computer-based language test. Nor do test development bodies make publicly available any information on how the interface for their computer-based language tests was developed. This article describes a three phase process model for interface design drawing on practices developed in the software industry, adapting them for computer-based language tests (CBTs). It describes good practice in initial design, emphasizes the importance of usability testing, and argues that only through following a principled approach to interface design can the threat of interface-related construct-irrelevant variance in test scores be avoided. The article also charts concurrent test development activities that take place during each phase of the design process. The model may be used in CBT project management, and it is argued that the publication of good interface design processes contributes to the mix of validity evidence presented to support the use of a CBT.

I Introduction

Although the number of computer-based language tests (CBTs) has grown rapidly during the last decade, there is little published literature on the process, or on good practice in CBT interface development or design for language tests. This is in stark contrast to the growing literature on commercial program design from companies like Microsoft (http://www.microsoft.com/usability/) and IBM (http://www-3.ibm.com/ibm/easy/eou_ext.nsf/publish/558) for whom poor design may result in poor sales and lost income, the numerous academic studies into the importance of interface design in human-computer interaction, and the work conducted on the Armed Services Vocational Aptitude Battery (ASVAB; Sands *et al.*, 1997). However, interface development and design is extremely important in computer-based language testing, where usability problems may constitute a threat to construct validity. In CBTs a poor interface design that is

Address for correspondence: Glenn Fulcher, Director, Centre for Applied Language Studies, University of Dundee, Dundee DD1 4HN, UK; email: g.fulcher@dundee.ac.uk

Language Testing 2003 20 (4) 384–408 10.1191/0265532203lt265oa

difficult to use for the test-taking population, or some important subgroup of the population, may easily become a source of construct-irrelevant variance, thus threatening the score users' ability to make meaningful inferences from test scores. Therefore, the careful documentation of key decisions supported by rationales and evidence contributes to the validity argument presented to support the meaning of test scores. This article presents a conceptual model for the design of a CBT interface, broken down into three distinct phases.

The first phase is planning and initial design, in which we consider putting together design teams, developing initial test specifications, and producing item and interface prototypes. When developing prototypes we look at a variety of issues, including hardware specifications and the principles of good interface design, such as navigation, text, page layout, terminology, help facilities, icons/graphics, the use of colour, and toolbars. The second phase begins once prototypes have been designed. The main activity in Phase II is usability testing, sometimes also referred to as the 'rapid iteration phase'. The primary purpose is to identify problems in the interface design and work on 'fixes' before the next iteration of testing. The selection and number of test-takers for each iteration, methods of data collection, and the monitoring of problems and fixes is discussed. In the third phase the developers move to field trials and fine tuning, using a test that more closely resembles the final product than the prototypes used in the usability studies. Field trials differ from usability studies in that they use a much larger group of test-takers. However, the role of the interface designer is much reduced in the final phase, limited only to fine tuning of any problems that may still exist and to which score variance may be attributed.

During the process of interface development other test development activities need to take place, but these are discussed only briefly to show how they may fit into an interface development model.

II The interface design process

The interface design process begins with essential planning activities, starting with the assembly of the multidisciplinary team required to deliver a working model of a computer-based test. Minimally the team should comprise applied linguists and language testers, designers, systems engineers, programmers and psychometricians. Language testers and applied linguists describe the test-taking population and test purpose. They also explicitly describe the constructs that the test is designed to measure. Once adequate descriptions are documented, they work on the content and item types that are hypothesized to best elicit from the test-takers responses that will provide evidence of

ability on constructs. The outcome of this activity is a working test specifications document, which forms the conceptual blueprint for everything that follows.

In Figure 1 below, it should be noted that work on a test interface cannot begin until there is a working test specifications document, and this is dependent upon the constructs the test is designed to measure having been defined and described. Construct definition and prototype item design are therefore included in Figure 1 as essential activities for interface design.

It is only when this essential work is complete and documented that work should begin on designing a prototype interface for the delivery of the items and content by members of the team with expertise in programming and computer systems. Initially, the design is based upon expert knowledge of design principles, which are discussed in more detail in Section III below.

When an operational prototype of the test has been designed, the process enters Phase II, often referred to as the 'rapid iteration' phase, or usability testing. Most frequently this consists of three separate tests of the interface, with the entire team working on 'fixes' to problems encountered at each step. When usability testing is complete, a near-final product is field tested with a larger sample from the test-taking population. The entire process is presented in Figure 1, and each phase is described in detail in subsequent sections.

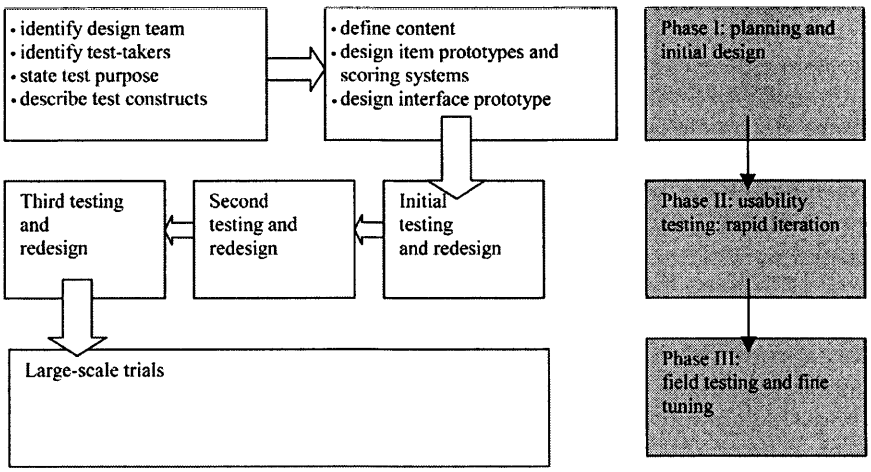

Figure 1 Essential components of a CBT interface design process

III Phase I: planning and initial design

1 Designing prototypes

This article does not discuss in detail the initial processes of describing the test-takers, the constructs to be measured, or arriving at initial content and item specifications. This is well documented in the language testing literature (Alderson *et al.*, 1995; Bachman and Palmer, 1996; Davidson and Lynch, 2002). What we are concerned with in this article is the process of turning the conceptual map into a computer interface that does not interfere with assessment. In other words, we are concerned with the mode of delivery and ensuring that it does not contaminate scores, thus threatening valid interpretation. The primary aim of good interface design is to reduce to a minimum construct-irrelevant variance that could be attributed to test method (Messick, 1989).

A prototype is a scaled-down preliminary version of a test interface. The main feature of a prototype is that it usually does not contain very much content. It may only have two or three examples of each of the item types that the test designers wish to include in the first form of the test. The purpose of the prototype is to allow usability testing before using valuable human and financial resources to build a complete test interface that may not work.

a Hardware considerations: The primary consideration is to design tests that will operate in the same way on any hardware configuration. However, this is extremely difficult to achieve.

- Computer specifications: A critical decision is the minimum hard disk size and computing speed for which the test will be designed. For example, if many of the intended test-takers are using older 486 or Pentium I machines, it is unwise to design a test that can only be taken on a Pentium III with 128mb RAM. This kind of information can only be obtained through initial studies into computer familiarity and availability.
- Screen resolution: The design should be for screen resolutions of 800 × 600 pixels, which is the most common resolution used. Higher resolutions (1024 × 768 and 1280 × 1024) result in small images on most screens, while lower resolutions (640 × 480) result in large images that could make it difficult to fit item content onto a single screen.
- Download time: In web CBT, download time is critical. Pages over 28k take longer than around 10 seconds to load on a 56k modem. Shneiderman (1984) and Martin and Corl (1986) have shown that with such waits for pages to load, users lose interest in the content.

b Software considerations:

- Browser compatibility: Many CBTs are now being designed for delivery through web browsers, whether they are delivered over the internet or local networks, or run from a CD ROM on a stand-alone machine. However, browsers are configured differently and there are significant differences between them. At the simplest level, it is possible for a designer to use and test a particular font size, and for this to be altered by the settings in the browser. Advice therefore needs to be provided to users on correct browser configuration. At the more complex level, java scripting differs in the two most popular browsers: Internet Explorer (IE) and Netscape. This is true, for example, in the embedding of audio or video files, and in the operation of buttons. Audio and video files appear differently in IE and Netscape, and the icons for play, forward wind, rewind and record are different. Autoplay commands in IE frequently do not work in Netscape, and vice versa. Buttons that work well in IE may not work in a version of Netscape Communicator lower than 4.6, and in earlier versions of either may not appear at all.
- Third party software: Many tests require third party software, sometimes referred to as plug-ins, to operate. This usually applies to tests that are designed for delivery through a web browser. In such pressing a button launches another program to play part of the test content. This is usually the case with video and audio files. However, tests that contain animation often require the test-taker's computer to have a program called Shockwave loaded. This is also required for items that involve the rearrangement of text, or moving text from one place to another on the screen. If a third party plug-in is used to make item types like this work, it is essential to provide the software with the test, or ensure that all machines are already loaded with the software.
- Authoring software: The selection of authoring software is dependent upon the item types and functionality that will be needed in the final product. The most popular authoring package for professional test designers is Macromedia Authorware (http://www.macromedia.com/), and Questionmark (http://www.questionmark.com/) is also widely used.

Once hardware and software decisions have been made, it is possible to proceed to the task of designing a test model.

2 Good interface design

The first step in good test interface design is the construction of a model or 'map' of what the product will look like, and what the routes

through the test are. This can usually be done using pencil and paper. It is then time to move into initial interface design. Many key considerations in interface design have been researched in the last decade, and generic guidelines are freely available (see Lewis and Rieman, 1994). The following issues are, however, specifically referenced to the computer-based interface for a language test.

a Navigation: Navigation is the term used to describe how the test-takers will move around the test. The most important things they will have to do are start the test, answer questions, move from page to page, and end the test. They usually do these tasks by clicking on buttons, although it may involve entering and submitting text, or operating the multimedia controls that play audio files. It should not be assumed that all test-takers are able to complete these tasks easily and automatically. The following guidelines should be adopted unless there is some reason to override them:

- Navigation buttons and icons should be kept to an absolute minimum. The user should not be confused with more options than are necessary at any point in the test.
- Operating system buttons and instructions should only be used where their function in the test is exactly the same as their function in the operating system. Users are already familiar with their common uses, and will be distracted by any unusual uses. Non-system test functions provide the interface designer with one of the first problems of initial design: developing new buttons and icons that have a clear meaning within the test, but do not resemble anything with which the user is already familiar. This leads to the problem of developing 'metaphors' (defined as iconic representations for navigation or events within the computer interface). Metaphors should be understandable to the test-taking population even if they are different from frequently used metaphors. The selection of appropriate icons is therefore a subject for research at the design phase (Dougall *et al.*, 2000). This is critical where the test-takers are from culturally diverse backgrounds, as it has been shown that there are culturally specific interpretations of icons across groups (Onibere *et al.*, 2001). This occurs when test-takers from different groups do not share the context from which a metaphor is drawn, and then cannot access the meaning of the icon through its representation on the screen (Bourges-Waldegg and Scrivener, 2000: 113). One example of this is that users from some cultures may not understand that a knife and fork (drawn from the context of airport restaurant signs) can represent the meaning 'list of restaurants' on a page for tourists. In a CBT a similar problem may arise by using an icon of an eraser to

represent the meaning 'undo your last response', or a question mark to represent the meaning 'help'. If the meaning cannot be retrieved by a particular group of test-takers, the functionality of the interface distracts the test-takers from focusing on the tasks and becomes a source of construct-irrelevant variance.

- Navigation should be quick and easy. One way to achieve this is to put all navigation elements in the same place on every page of the test, so that test-takers do not have to continually work out how to proceed.
- Each page should have a clear title at the top of the page that relates to a map of the test. The test-takers should never be 'lost' in the system, but know exactly where they are. Clear titles have been shown to improve human–computer interaction (Gerhardt-Powals, 1996; Levine, 1996) and, in a test-taking environment, relating these to a map of the test provides a sense of security and control which is otherwise lacking.
- Within the map of the test a critical decision is whether to allow test-takers to 'revisit' questions/pages, or permanently remove them once a question has been attempted or a page viewed. In adaptive tests it is not possible to allow revisiting, although in linear tests it is more feasible. If revisiting is allowed, the navigation must be more complex, not only to allow movement around the entire test, but also to return the test-taker easily to the point at which he or she started to revisit.
- If revisiting is not allowed, it is essential to minimize the possibility that test-takers can make mistakes. Examples of this include moving on to another page accidentally, submitting an answer they did not intend to, or not providing an answer to an item before moving to the next one. Ironically, reducing the possibility of unintended moves may sometimes lead to the introduction of more steps in the process, such as adding an additional pop-up box that warns the test-taker of the consequences of continuing, and asks for confirmation that this is indeed what they wish to do. The interface designer must try to balance the cost of adding further steps against the benefits of simplicity.

If the navigational structure of the test is unclear, requires undue attention, or leads to test-taker errors, it is arguably the case that the scores will be meaningless.

b Terminology: Terms used in instructions should be clear, simple and consistent. Good practice is to draw up a list of 'reserved words' that will be used to direct the test-taker to perform certain actions. One example might be 'OK' – which should mean 'accept the entered answer and proceed.' Another may be 'cancel', which might be used

to mean 'delete my answers, as I wish to try again.' Reserved words should be used consistently, and no words that do not occur in the reserved word list should be used in the interface.

c Page layout: The most important issue in page layout is the amount of information that is presented on a single screen. Too much detail makes it difficult for test-takers to distinguish between critical elements of the test task and information that is recurrent or ancillary to the test task.

- Pages should not be overcrowded with information. The page design should draw the attention of the test-taker to the important tasks, which may require the use of empty white space. Research into information density on web pages has to date found that dense information is more easily scanned, and is more appealing, than pages that use empty space (Staggers, 1993; Spool *et al.*, 1997); However, no similar research appears to have been conducted in relation to CBT interfaces. This is an area that would benefit from further investigation. Whatever the optimum information density, it is important to ensure that navigational aids or other non-test information is separate and distinct from critical tasks. All recurring navigational aids should occur in the same place on every page, so that test-takers do not have to re-read non-critical information as they navigate through the test.
- In reading items scrolling should be reduced to a minimum (maximum two screens). Readers unfamiliar with scrolling may spend up to 13% of their time moving up and down the text (Dyson and Kipping, 1998).

If a range of item types are to be used in a test, it is inevitable that page layout will look somewhat different as the test-taker proceeds through the test. However, every care should be taken to harmonize the layout and 'feel' of the pages as much as possible. The primary principle is that no page should come as a surprise to any test-taker: familiarity of format from page to page aids test-takers in anticipating response requirements, which reduces their need to attend to non-task elements.

d Text: Most of the instructions and test tasks require the use of text. All text should be clear on the screen and easy to read for all test-takers. Text should not be manipulated for presentational purposes in such a way that it interferes with the test-takers' ability to read at their normal speed.

- Avoidance of upper case text is strongly advisable in CBTs,

especially for instructions. In computer communication the use of upper case text has become associated with simulated shouting, and for many test-takers the impression given is that the system is angry. Even for those who are not familiar with this use, upper case text is more difficult to read than normal text.

- Similarly, the use of moving or animated text should be avoided. Not only is it difficult to read, but it also distracts the test-taker from any other information presented on the screen.
- A font size larger than 10 point, preferably 12 point or higher, should be used. Tullis *et al.* (1995) have shown that font sizes below 10 point reduce reading speed on a computer interface.
- An early decision should be taken regarding whether test-takers should be allowed to alter the font size to one with which they are comfortable. This may provide an advantage to any test-takers who are visually impaired. However, it does provide an additional option for other test-takers that could distract them from the test and introduce another source of construct-irrelevant variance. Another problem is that pages may become longer on some monitors. This could introduce the need to scroll through text which was designed to be presented on a single page without the need to use a mouse. Given the threat to construct validity that this kind of accommodation may pose, designers must make informed decisions about the relative costs and benefits of introducing another option.
- It is advisable to use a familiar font such as Times Roman or Arial, which are the preferred default options for most operating systems. Unusual fonts, however attractive they appear, should be avoided, as they can slow down the reading speed of many test-takers.
- Fonts should not be mixed in a test interface. While it is tempting to do this to draw a distinction between titles and instructions, or instructions and items, different fonts may interfere with the test-takers' ability to read.

e Colour: The use of colour should attract the user's attention to the most salient part of the task, but task completion should not depend upon the test-taker's ability to understand colour coding. The most common problem encountered is with users who cannot distinguish between red and green. Also problematic is any combination of colours that causes eye strain or headaches, such as blue or black text on a yellow background.

It is important to take into account any test-takers who may be visually impaired. The easiest way to avoid problems is to use colours that contrast highly, like black text on a white background rather than

red on an orange background (Chisholm *et al.*, 1999). The most important aspects of designing a colour scheme for a test interface are:

- hue: the perceptual attribute of the basic colours (blue, green, yellow and red);
- lightness: how much light appears to be reflected from the surface of the colour;
- saturation: the degree of colour intensity.

The basic rule in colour design is always to maximize contrast between colours, by differentiating between hue, lightness and saturation.

f Toolbars/controls: As we have seen, the use of metaphors in toolbars can help test-takers transfer existing computer knowledge to the test environment. An example is the image of traffic lights to show that the test-taker can proceed to the next item or must stay on the present item because it is not yet complete. Little research has been conducted into the number of items on a toolbar that test-takers can reasonably manage during a test. The fewest the designers can use should be the default option.

When using text in controls it is essential to keep sentences short and use words that are much easier to understand than lexical items that occur in the test items or texts. The most important information should come first in any textual guidelines.

g Icons/graphics: Icons and graphics are used to develop metaphors. The design or selection of icons involves not only the conceptual problems discussed above, but technical and presentational issues. General guidelines on the use of icons and graphics include the basic rules which follow.

- The number of icons to be used should be kept to an absolute minimum: use only those that are essential to the operation. Graphics should never be added simply to make a page more attractive. There is a problem with commercial test packages, as marketing requirements usually require branding, with the inclusion of logos, background images or company design schemes. The impact of these designs is rarely tested.
- The size of icons or graphics is particular important. While they must be clearly seen by the test-taker, they should not be so large that they take up too much space on any page. If the test is designed for internet delivery, the size of icons will seriously impact on download time.
- Animated or blinking images should be avoided as these distract the test-taker from other content on the page.

- If there is a time delay in any part of the test (as may be the case with loading audio or video files), it is important that test-takers see an icon that indicates what is happening, and how long they may have to wait. The metaphor normally selected for this is the egg timer. A failure to include this feature may lead some test-takers to think that the program has crashed; the result is that they either panic or start pressing keys to get the software to work again. In the worst instance a test-taker may inadvertently cause the program to crash or reboot the machine, unless the test is being taken in a controlled environment where the designers can disable the keyboard. Green (2000: 31) also argues that a timer needs to be clearly visible on the screen if the whole test, or parts of the test, have to be completed within a given time limit, even though timing studies have shown that most test-takers are able to complete the tasks within any time limits set.

Special care needs to be taken when designing and positioning any icons that indicate that the test-taker has finished an item or the entire test, as these may trigger an event to save the responses to disk, or submit the responses to the scoring program. Triggering non-redeemable events unintentionally must be avoided.

h Help facilities: Help facilities allow a test-taker to step outside the test at any point in order to seek clarification on how to proceed. A general help facility, such as those provided with commercial word processing software, allows the user to call up a searchable help document. While this may be an attractive idea in a testing context, it should be remembered that unless the help facility is available in the test-taker's native language it is difficult to write the help in such a way that it is quick and easy to process. Further, stepping outside the test in this way reduces the time available for answering test items. It is preferable to design the interface, navigation and instructions in such a way that a help facility is not required. A test interface should not be as complex as word processing packages.

Contextual help is a useful alternative where, for example, a user may drag a question mark icon over a test item. A dialog pop-up box may then explain how to answer the item. However, in principle it is better to solve potential problems with answering items through good item design and ensuring that all test-takers are familiar with the test format prior to operational testing.

Nevertheless, it should not be assumed that all test-takers will navigate around the system easily, or do precisely what the designers intend. One example that has already been mentioned is moving from one test item to another without responding. In such cases the program needs to provide a clear 'error message' that indicates to the

test-taker what the mistake is, and how to correct it. The language of the error message needs to be short and simple. The programmers should attempt to think of all possible errors that a test-taker may make (often discovered during usability studies) and provide a way out.

i Stepping outside the test: All CBTs are self-contained. That is, they do not allow the user to step outside the test to consult information that may be stored in a remote location. It is feasible, however, to design a test that requires the test-taker to use information on the internet to answer test items. An example might be to allow test-takers to access a range of texts to answer 'reading to learn' items, as described in Enright *et al.* (2000), which would involve combining information from a range of sources to answer the item correctly. A complex construct definition such as reading to learn might lead to the introduction of test content that encourages more freedom in movement from the test interface to the internet and back, perhaps mirroring how internet users actually use the web for reading. If such novel tasks are used, it would be essential to conduct research to discover what impact they have on test-takers, scores and score meaning.

j Selecting item types: The selection and use of item types depends upon the initial construct definition for the test, and an evaluation (rationale and empirical evidence) that prototype items elicit evidence to support appropriate construct inferences. While it is advisable to use a range of item types, it is also possible to use so many types that the test-takers are constantly required to use different computer skills to respond. There is no published research into the optimal number of item types, or the impact of mixing item types within sections of tests. The best guidelines are therefore experiential, suggesting that no more than seven or eight item types should be used in any single test. The most popular item type remains multiple-choice because it is the easiest to score using current Item Response Theory (IRT). However, it manifests itself in many forms, from traditional four-option multiple-choice items to embedded cloze items with the options contained in a pull-down menu. With the latter item type care needs to be taken as the menu can obscure important parts of the text. Research is needed to ensure that this does not affect how learners at different levels of reading ability process the text.

k Use of multimedia: The inclusion of multimedia is one of the most exciting areas of future development of CBTs, but the introduction of video files in tests has been slow because of uncertainty in

how it may impact upon the ways in which test-takers process sound with pictures (see Ginther and Chawla, 1997). However, from an interface design perspective there is little problem with the inclusion of such files as long as presentation is consistent, and loading times are reasonably short. CBTs have currently only made extensive use of audio files for listening tests. The design of the play, stop and volume controls is of most importance, and test-takers should be given practice in using these prior to attempting items that are going to be scored.

More problematic is likely to be implementing a computer-based test of speaking. Apart from systems and file size/storage considerations, the test-taker is likely to be asked to: read instructions, read the prompt, prepare to speak (thinking time), start speaking, and stop speaking. It is most likely that the interface will be designed so that interaction between the test-taker and the computer is minimal, or non-existent. Timing studies should be conducted to discover how much time is to be allocated to the first three steps so that recording begins automatically, and the start of recording is clearly indicated to the test-taker. Similarly, the amount of time remaining before recording ends needs to be displayed, and the end of recording clearly communicated to the test-taker. Speaking involves greater cognitive load and use of short-term memory than closed-response items; attending to the interface is therefore more likely to impact upon performance and introduce construct-irrelevant variance. This is an area that requires extensive research.

l Forms for writing/short answer tasks: Many CBTs require the test-taker to type a response to an item. These may be single word answers that are matched against a template for automatic scoring, or short sentences and extended pieces of writing that need to be stored for future human scoring. In these tasks the interface designer needs to ensure that:

- there is enough space to type the required response;
- any multiple text entry boxes are aligned or justified;
- multiple boxes are arranged vertically rather than horizontally.

Text entry boxes that are misaligned or arranged horizontally create an uneven page layout that is difficult to navigate, as the test-taker must click on the boxes to begin text entry (Tullis, 1983). The exception to this might be a cloze test, in which the text boxes are set within a reading passage. Such item types require careful study in Phase II usability testing to establish that all test-takers were able to manipulate the mouse sufficiently to ensure that multiple clicking in non-aligned boxes does not create a problem.

A special case of text entry problem relates to the collection of personal details from the test-taker, rather than collecting short answer or writing samples. Most CBTs require the test-taker to enter (at least) their name and identification number. The design of the format for the entry of this data should be clear, unambiguous, and easy to complete.

m Feedback: It is likely that a CBT will provide some form of feedback to the test-taker. This may simply be an indication that the test has finished, and require a form of words that terminates the test in such a way that ensures the test-taker leaves with a positive experience of the computer–human interaction. However, if scoring is automatic and immediate, feedback may include a score and a verbal description of language level in relation to the construct definitions. While the precise nature of the feedback cannot be determined until the third phase of interface design, a preliminary decision should be taken about the intended form and amount of feedback in Phase I, so that an appropriate page can be developed, even if this initially lacks detailed content. The feedback page should form part of the Phase II usability testing so that test-taker reactions to the way the test is concluded can be evaluated.

3 Phase I: concurrent activities

During Phase I there is no 'test' as such, only a developing prototype containing a number of new items that are created by the applied linguists working from the construct definitions to create initial test specifications. However, there are a number of other test development activities that must run concurrently. These include:

a The development of delivery systems: These may be local area networks (LANs), internet systems or stand alone systems. In the latter case facilities to store score data to disk should be considered.

b The investigation of score retrieval and database storage: Scores need to be passed from the machine to a secure database over a LAN or the internet. One important question is whether the final score is calculated on the local machine, or processed by the central database. If the latter option is chosen, a further question is whether (and how) the score will be sent back to the test-taker and/or another score user.

c Distribution and retrieval systems for any sections to be scored by humans: If the test includes constructed-response items, or if any speech samples have been collected, these need to be distributed to human raters for scoring. The human raters also should be able to

submit the scores for constructed-response items to the database electronically.

d Scoring algorithms (and rubrics if necessary): As soon as prototype items are available, psychometricians need to begin work on scoring algorithms. This is a critical step, because decisions relating to scoring will have a direct impact upon the interface design. For example, if the CBT is to be adaptive, it may become essential to present a single item on each page. In a reading test, however, if it is decided to work at the level of testlet adaptivity (see Wainer and Kiely, 1987), multiple items may be presented on a single page, with branching to other pages depending on the response to the testlet. Such decisions are also influenced by systems engineers, especially if the test is automatically scored on a remote server. If a test is to be adaptive at the item level, will it be possible for the remote system to accept the simultaneous number of submissions likely to be received from the volume of test-takers expected to be taking the test at a given time? Systems considerations such as these may force the CBT developer to consider the use of linear forms that are easier to process. Finally, if constructed-response items or speech samples are to be elicited, the scoring rubrics need to be devised. These may be altered during Phase II and Phase III, but should be constructed in prototype form at the same time as the items are being created.

e Familiarity studies: It is essential to conduct studies into the familiarity of the test-taking population with computer systems and software. These studies will inform the development of prototype metaphors and navigation systems, as well as forming the basis for tutorial development in Phase III (see, for example, Kirsch *et al.*, 1998; Taylor *et al.*, 1998).

f Available technology studies: Similarly, it is important to know the specification of hardware available to the test-taking population. This information will inform decisions regarding the minimum specifications for which the test may be written, the feasibility of internet delivery and the file sizes that might be used in the test. It will also inform decisions about the third party plug-ins that might be used, or whether the test can be delivered through conventional browsers.

g Initial construct validity studies: Once a number of prototype items are available, initial construct validity studies may be conducted. These are unlikely to be quantitative, but may involve expert feedback from language teachers and testers about whether the items developed are likely (in their opinion) to elicit evidence to support

the hypothesized inferences. Feedback of this nature at an early stage in the project may aid the process of reducing, increasing, or refining the initial pool of prototype items.

h Small-scale item trialling: Item and task prototypes should be piloted with small groups of users in order to make decisions about which are to remain in the CBT and which are to be excluded.

When a prototype interface with a small pool of items is complete, the process moves into the second phase of usability testing.

IV Phase II: Usability testing

1 Searching for problems and solutions

In usability testing a small number of test-takers, usually alone or in pairs, undertake a number of prototype items or tasks, within the prototype interface. The test-takers are observed by one or two designers, who make notes on the observable performance of the test-takers and conduct a debriefing interview at the end of the process. Frequently test-takers are encouraged to 'think aloud' as they are doing the test and the observers record the protocols for later analysis (see Green, 1998). Observers may also interrupt test-takers to ask questions if they see the test-taker facing undue difficulty in navigating the interface or completing tasks.

The observers are looking for specific features of the interaction between the test-takers and the interface/prototypes, such as:

- Are the test-takers able to navigate easily from one item or page to another?
- Can they easily respond to each prompt?
- Is the speed at which they are able to work appropriate?
- Do they stumble or become confused? If so, why?

One description of a usability study, in the context of the ASVAB, considered legibility of test items, comprehension of instructions, effects of fatigue and test administration factors (e.g., clock time), as well as familiarity issues (Vincino and Moreno, 1997).

After the observation stage structured interviews are conducted with individual test-takers, which may lead to a structured discussion with a focus group: a small number of test-takers who are encouraged to present their reactions to the interface. These interviews and discussions are normally recorded for later analysis.

At the end of the session the observers extract problems encountered by the test-takers from their notes, the think-aloud protocols and debriefing interviews/focus group tapes. These are prioritized

according to the seriousness of the problem and the frequency with which it was encountered. This qualitative data analysis generates a list of potential problems that becomes a list of action points for the revision of the interface.

The following is a typical method of categorizing the data from interviews, think aloud protocols, interviews and focus groups (adapted from Sullivan, online).

a Problem Identification: Problems are rated with one of three levels of severity:

- Level 1: Users were unable to continue with a task or series of tasks due to the problem.
- Level 2: Users had considerable difficulty completing a task or series of tasks but were eventually able to continue.
- Level 3: Users had minor difficulty completing a task or series of tasks.

b Problem resolution:

- Addressed: The problem is fixed and successfully used with a new group of users.
- Partially addressed: A fix is implemented and tested with users; while satisfactory, some issues remain to be addressed.
- Planned: A fix for the problem is designed but has not been implemented.
- Undecided: There is uncertainty whether to fix the problem or it is unclear if a fix is feasible.
- Not addressed: The problem is not going to be addressed as it is minor and/or there is not enough time or resources to address the problem.

Another commonly used system for rating problems is provided by Nielsen (1994):

c Problem identification:

- The frequency with which the problem occurs: Is it common or rare?
- The impact of the problem if it occurs: Will it be easy or difficult for the users to overcome?
- The persistence of the problem: Is it a one-time problem that users can overcome once they know about it or will users repeatedly be bothered by the problem?

d Problem resolution:

0 = I don't agree that this is a usability problem at all.

1 = Cosmetic problem only: need not be fixed unless extra time is available on project.
2 = Minor usability problem: fixing this should be given low priority.
3 = Major usability problem: important to fix, so should be given high priority.
4 = Usability catastrophe: imperative to fix this before product can be released.

The usability testing of a CBT interface is an iterative process that leads in each stage to improvements that reduce the problems encountered by new groups of test-takers. The tracking of usability problems, their seriousness and the fixes, is extremely important in Phase II. This requires the transfer of information to a database, so that in each iteration the number of remaining and new problems can be charted.

At what point the iterative process ends and the prototype interface becomes 'fixed' is a matter of judgement related to the decreasing number and seriousness of problems encountered in each iteration, but three iterations is common before the test is subjected to a large-scale field test.

2 Selecting test-takers for usability studies

The test-takers should:

- be representative of the test-taking population;
- have a range of experience of computer interfaces;
- have a variety of expectations of a CBT;
- represent the different levels of language ability to be tested.

This information is drawn from the familiarity and availability studies and the description of the test-taking population conducted during Phase I.

The next question concerns the number of test-takers that are needed for a usability study. Nielsen and Landauer (1993) conducted a metastudy of the number of usability problems generally encountered in the usability literature, and suggested the following formula for deciding on the number of subjects:

$$N(1 - (1 - L)^n)$$

N is the total number of usability problems encountered in a study, and L is the proportion of usability problems discovered using a single subject. Across all the studies that Nielsen and Landauer reviewed, they found that the typical value of L is 31%. They plotted this on a curve, shown in Figure 2.

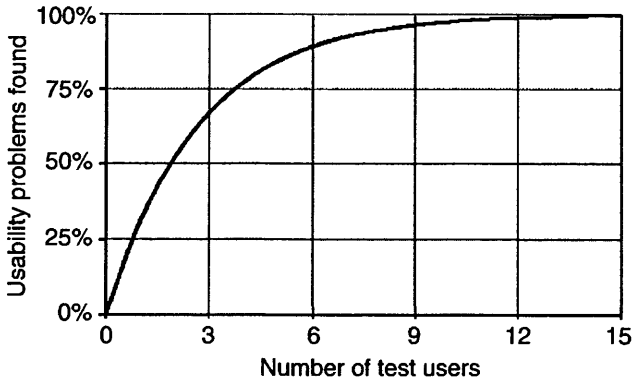

Figure 2 Usability problems and number of participants
Source: Reproduced by permission of Jacob Nielsen from Alertbox, March 2000; available at http://www.useit.com/alertbox/2000319.html

Nielsen and Landauer argue that the first participant provides most of the insights to usability problems of an interface design. The second participant provides more information, but the overlap with the first participant reduces the proportion of new information. The same is true of each additional participant. They recommend limiting the participants to five because after this the addition of new information is minimal and not worth the expenditure. Further, using multiple studies of five participants ensures that the process of interface design is iterative. After the first study the test interface design can be improved and tested again.

It is therefore advisable to run three separate usability studies, each using five persons, with interface revision after each study. This is the most likely procedure to identify and eradicate most of the usability problems encountered. However, it should be noted that this assumes a homogenous target population for the test, from which the participants can be drawn. If there are distinct groups of potential test-takers within the target population, it is necessary to conduct discrete usability studies for each identifiable group within the population.

3 Phase II: concurrent activities

Phase II can be seen as working from the initial design to a final product that can be taken for field testing. While the interface is being

refined, a number of critical activities must run concurrently if a CBT is to be ready at the end of the process.

a Item writing and banking: It will be remembered that the prototype used in the usability studies is a much reduced version of the final product. Following small-scale item trialling and taking final decisions about which item types will appear in the final test, large-scale item writing needs to be undertaken. If the CBT is to be an adaptive test, the number of items generated needs to be exceptionally large. Items need to be banked in the format that they will be required in the operational version of the test.

b Pre-testing: In Phase I small-scale item trials were conducted to select those that would go into Phase II. In the second phase they need to be pre-tested on much larger groups of test-takers in order to establish item parameters. Items from the bank that do not exhibit the characteristics required are removed at this stage.

c Trialling scoring rubrics: If constructed-response items are to be used, a small number of raters (10–15) are trained in using the scoring rubrics developed in Phase I. Samples are collected in the pre-testing and sent for rating. Inter- and intra-rater reliability is calculated, and debriefing interviews conducted to discover if there are problems with the rubrics. The scoring rubrics are refined as needed.

d Structural construct studies: If the test is designed to measure more than one construct, it is possible at this stage to begin investigating the relationships between the different parts of the test using correlational and related techniques.

V Phase III: field testing and fine tuning

1 The purpose of field testing

The role of the interface designers is minimal in Phase III. The interface should not undergo any major changes once a test is ready for field testing with a large sample drawn from the test-taking population. Field testing is primarily required for scaling the test and ensuring that the logistics of data collection, submission, scoring, distribution and retrieval, and feedback work as planned. Field testing is a dry run of the complete system prior to making the test operational, and the interface is not itself under separate scrutiny. It does,

however, provide an opportunity to test for variation in the appearance of the interface across sites, machines and platforms.

2 Concerns of the interface designers in field testing

The interface designers are primarily concerned to collect any evidence that may point to interface-related score variance only if this is a possibility because of any remaining problems that have not been fixed in the usability studies. If such concerns exist, studies using ANOVA or related techniques may be used to investigate whether remaining interface problems prove to be significant. Fine adjustments to the interface may be made if necessary.

3 Phase III: concurrent activities

Phase III concurrent activities are directed towards the production of the operational version of the test, and providing meaningful scores to score users. It should also prepare the way for further validation activities beyond the first operational use of the test.

a Developing tutorials: Although every care should have been taken through the familiarity, availability and usability studies to reduce the problems that test-takers may encounter, it should not be assumed that every test-taker will have the skills and knowledge required to complete the test as expected. Tutorials need to be developed that explain and give practice in the key skills required (such as scrolling and the use of the mouse), and the navigation system/metaphors used in the interface. A tutorial may be included at the beginning of the test, as in the Test of English as a Foreign Language, or made available as a separate package to the test-taker prior to a testing session. Green (2000: 30–31) suggests that test-takers should not be allowed to proceed to the test unless they complete the tutorial, in cases where the system has to repeat the instructions many times, and where the test-taker cannot correctly answer very simple demonstration items.

b Producing practice/example tests: Practice tests need to be constructed to familiarize test-takers with all the item and task types included in the test.

c Rater training packages and rater training: If constructed-response items are used in the test, a rater training package needs to be developed, including the scoring rubrics and sample answers from

the field testing. The package is then used to train the requisite number of raters to score the samples from operational tests.

d Scaling studies and score reporting: Scaling studies provide the means to report meaningful numerical scores to test-takers and score users through establishing the statistical properties of the test and the range of scores obtainable. However, it is becoming more common to provide a verbal descriptor of what the scores mean in terms of what the test-taker is capable of doing in the language with a score at a given level on the scale. Verbal descriptors may be generated during and immediately after field testing by relating scale scores to the initial construct definitions developed during Phase I. Such descriptors will become the subject of further validation studies.

e Planning further validation studies: Prior to the operational use of the test further validation studies may be planned, such as the meaningfulness of the verbal descriptors attached to score ranges. They may also include concurrent validity studies with scores on other tests, or tutor ratings, for those taking part in field testing. However, further validation studies to be conducted after the operational launch should be planned at this stage with the purpose of collecting evidence that will strengthen score interpretation over a more extended period of time. This acknowledges that validation is an ongoing process.

VI Conclusions

This article has discussed the process of designing an interface for a computer-based test. It has presented a process model broken down into three distinct phases. Phase I is planning and initial design, in which account must be taken of good practice in the design of navigation systems, metaphors and the presentation of test content. In Phase II the interface is tested for usability in an iterative process with constant revision. In Phase III there may be fine tuning to the interface as a result of larger-scale field testing. Throughout the process the aim of good interface design is to make the interface easy and quick to use for the test-taker so that it does not constitute a source of construct-irrelevant variance, thus threatening the inferences that may be drawn from test scores.

It is argued that the model presented in this article may be used in a principled approach to interface development for computer-based tests in the future. Test producers are encouraged to publish details of the process of interface design. In demonstrating the care

that has been taken to minimize the impact of interface construct-irrelevant variance, such publications would form part of an important mix of validity evidence available to support test use and interpretation.

VII References

Alderson, J.C., Clapham, C. and **Wall, D.** 1995: *Language test construction and evaluation*. Cambridge: Cambridge University Press.

Bachman, L.F. and **Palmer, A.S.** 1996: *Language testing in practice*. Oxford: Oxford University Press.

Bourges-Weldegg, P. and **Scrivener, S.A.R.** 2000: Applying and testing an approach to design for culturally diverse user groups. *Interacting with Computers* 13, 111–26.

Chisholm, W., Vanderheiden, G. and **Jacobs, I.**, editors, 1999: *Web content accessibility guidelines 1.0*, http://www.w3.org/TR/WCAG10/WAI-WEBCONTENT-19990505/ (accessed May 2003).

Davidson, F. and **Lynch, B.K.** 2002: *Testcraft: a teacher's guide to writing and using language test specifications*. New Haven, CT: Yale University Press.

Dougall, S.J.P., de Bruijn, O. and **Curry, M.B.** 2000: Exploring the effects of icon characteristics on user performance: the role of icon concreteness, complexity and distinctiveness. *Journal of Experimental Psychology* 6, 291–306.

Dyson, M. and **Kipping, G.** 1998. The effect of line length and method of movement on patterns of reading from screen. *Visible Language* 32, 181.

Enright, M., Grabe, W., Koda, K., Mosenthal, P., Mulcahy-Ernt, P. and **Schedl, M.** 2000: *TOEFL 2000 reading framework: a working paper*. Princeton, NJ: Educational Testing Service. Available online at: ftp://ets.org/pub/toefl/253718.pdf (accessed May 2003).

Gerhardt-Powals, J. 1996: Cognitive engineering principles for enhancing human-computer performance. *International Journal of Human-Computer Interaction* 8, 189–211.

Ginther, A. and **Chawla, A.** 1997: Multimedia: words with pictures: unpacking the effects of visual accompaniments to listening comprehension items. In Huhta, A., Kohonen, V., Kurki-Suonio, L. and Luoma, S., editors, *Current developments and alternatives in language assessment*. Jyväskylä, Finland: University of Jyväskylä.

Green, A. 1998: *Verbal protocol analysis in language testing research*. Cambridge: Cambridge University Press.

Green, B.F. 2000: System design and operation. In Wainer, H., editor, *Computerized adaptive testing: a primer*. Second Edition. Mahwah, NJ: Lawrence Erlbaum, 23–35.

Kirsch, I., Jamieson, J., Taylor, C. and **Eignor, D.** 1998: *Computer familiarity among TOEFL examinees*. TOEFL Research Report 59.

Princeton, NJ: Educational Testing Service. Available online at: ftp://ftp.ets.org/pub/toefl/275755.pdf (accessed May 2003).

Levine, R. 1996: *Guide to web style*. Santa Clara, CA: Sun Microsystems.

Lewis, C. and **Rieman, J.** 1994: Task centered user interface design: a practical introduction. Available online at: ftp://ftp.cs.colorado.edu/pub/distribs/clewis/HCI-Design-Book/ (accessed May 2003).

Martin, G.L. and **Corl, K.G.** 1986: System response time effects on user productivity. *Behaviour and Information Technology* 5, 3–13.

Messick, S.A. 1989: Validity. In Linn, R.L., editor, *Educational measurement*. 3rd edition. New York: American Council on Education/Macmillan Publishing Company, 13–103.

Nielsen, J. 1994: *Usability engineering*. San Francisco, CA: Morgan Kaufmann.

Nielsen, J. and **Landauer, T.K.** 1993: A mathematical model of the finding of usability problems. *Proceedings of ACM INTERCHI'93 Conference*, Amsterdam, April, 206–13.

Onibere, E.A., Morgan, S., Busang, E.M. and **Mpoeleng, D.** 2001: Human–computer interface design issues for a multi-cultural and multi-lingual English speaking country: Botswana. *Interacting with Computers* 13, 497–512.

Sands, W.A., Waters, K.B. and **McBride, J.R.**, editors, 1997: *Computerized adaptive testing: from inquiry to operation*. Washington, DC: American Psychological Association.

Shneiderman, B. 1984: Response time and display rate in human performance with computers. *Computing Surveys* 16, 265–85.

Spool, J.M., Scanlon, T., Schroeder, W., Synder, C. and **DeAngelo, T.** 1997: *Web site usability: a designer's guide*. North Andover, MA: User Interface Engineering.

Staggers, N. 1993: Impact of screen density on clinical nurses' computer task performance and subjective screen satisfaction. *International Journal of Man-Machine Studies* 39, 775–92.

Sullivan, K. *The Windows® 95 user interface: a case study in usability engineering*. Available on-line at http://www.acm.org/sigchi/chi96/proceedings/desbrief/Sullivan/kds_txt.htm (accessed May 2003).

Taylor, C., Jamieson, J., Eignor, D. and **Kirsch, I.** 1998: *The relationship between computer familiarity and performance on computer-based TOEFL test tasks*. TOEFL Research Reports 61, Educational Testing Service, Princeton NJ. Available online at: ftp://ftp.ets.org/pub/toefl/275757.pdf (accessed May 2003).

Tullis, T.S. 1983: The formatting of alphanumeric displays: a review and analysis. *Human Factors* 25, 657–82.

Tullis, T.S., Boynton, J.L. and **Hersh, H.** 1995. Readability of fonts in the windows environment. *Proceedings of Special Interest Group on Computer Human Interaction, 1995*, 127–28. Available on-line at http://www.acm.org/sigchi/chi95/proceedings/intpost/tst_bdy.htm (accessed May 2003).

Vincino, F.L. and **Moreno, K.E.** 1997: Human factors in the cat system: a pilot study. In Sands, W.A., Waters, K.B. and McBride, J.R., editors,

Computerized adaptive testing: from inquiry to operation. Washington, DC: American Psychological Association, 157–60.

Wainer, H. and **Kiely, G.** 1987: Item clusters and computerized adaptive testing: a case for testlets. *Journal of Educational Measurement* 24, 185–202.