

Usability and Voting Technology
Frederick G. Conrad
Bureau of Labor Statistics

White paper for Voting Technology Workshop

The success of voting technology depends, ultimately, on voters' ability to use it. A particular voting system may be secure, stable and computationally flawless but if voters are stymied by the user interface (the point of contact between the system and the voters) the system may fail to provide an accurate assessment of the voters' will. The costs associated with systems that are hard and frustrating to use are increasingly (though not universally) recognized by the software development community and, in response, a collection of techniques and activities known as usability engineering or user centered system design is becoming a standard part of the development process. This white paper explores the application of these usability ideas and techniques to voting technology. The next section introduces basic usability concepts. This is followed by a section on the mechanics of evaluating a system's usability – usability testing. The final section considers the kinds of usability issues that are specific to voting technology and considers how one might detect such usability problems early enough to fix them.

What is usability?

The usability of a piece of software is usually measured in terms of accuracy, speed and satisfaction of use. Accuracy can be thought of as the overall outcome of a task (e.g. in a voting system, accurate use might be defined as successfully registering a vote for the intended candidate) or the smaller steps along the way (e.g. positioning the mouse cursor on a checkbox with enough precision that the click is interpreted by the system as an attempt to check the checkbox). The same distinction can be drawn for speed of use: overall time to complete the task or time to perform a component task. Psychologists have long been concerned with speed-accuracy tradeoffs (completing the task more quickly can lead to more errors) and so, in usability tests, it is important to instruct users clearly on how they should balance the two. User satisfaction is typically measured with a questionnaire after the user has interacted with the software. One can administer off-the-shelf satisfaction questionnaires or can customize a questionnaire for the particular software product. The advantage of off-the-shelf questionnaires (at least the good ones) is that they have been demonstrated to be reliable; the down side is that they may solicit satisfaction judgments at too general a level to help improve specific aspects of the design. Customized satisfaction questionnaires have the opposite characteristics: no measure of reliability but they can assess user satisfaction of very specific system features.

The time and expense of usability-related activities (primarily testing end-users in laboratory settings) is usually justified in terms of cost savings (e.g. Bias & Mayhew, 1994) and, thus, increased profit. However, when profit is not the goal, advocates of user-centered design must appeal to other benefits of usable systems. For example, when the accuracy of users' actions – and the thinking behind those actions – is a priority, as in answering questions on computerized Federal surveys – the cost of usability testing can

be justified in terms of improved data quality. For example, if web surveys enable respondents to obtain a definition for a term in a question by taking a simple action like clicking a link, respondents are more likely to interpret the question as its author intended and, as a result, more likely to answer accurately. The result is better survey estimates of the population. Voting is similar. There is no profit rationale for the various governments who conduct elections. Instead, the rationale for usable voting technology is that it promotes accurate measurement of the public's will. In addition, usable voting systems are more likely to promote participation than unusable systems. If voters are not highly motivated to participate in the first place, a frustrating experience interacting with the system may well push them over the non-participation threshold.

What is Usability Testing?

Building usable systems needs to begin prior to the creation of any software. In particular, the users' task must be analyzed. Essentially this consists of determining the users' goals and the necessary steps to achieve those goals. In principle, the components of the task should drive the design of the user interface; components of the task are mapped to possible screen objects and the objects are configured as clearly and usefully as possible given what the designer knows about the task. This kind of upfront analysis is particularly important for tasks the user carries out in the absence of the to-be-developed software (as in voting where voters currently use paper ballots, mechanical machines, etc.) because the introduction of the software may well change the components of the task.

Assuming that the designers have conducted a task analysis, and that this has informed the preliminary design of the system, the emphasis then shifts to usability testing. This is usually understood to mean experimental sessions in which plausible end users interact with a prototype version or paper mock-up of the software and this interaction provides empirical evidence of key usability problems (as well as successes). The outcome of the tests is often just a list of problems with the current version of the software, in which problems involve some kind of inaccurate action by the user or indication of user uncertainty. The latter may be indicated by the users' inaction (reflected in slow completion of the task) or a comment by the user. Such a list is more compelling when accompanied by video examples of users experiencing the listed problems. Sometimes the problems are quantified with measures such as their frequency of occurrence or mean duration.

The results of usability test sessions are easiest to interpret if users are assigned specific tasks with clear correct and incorrect outcomes. This makes it straightforward to assess overall accuracy. If there are specific sequences of actions the user must take to achieve the correct outcome, then accuracy can be assessed at a more micro level. Even if there is not a correct result, assigning multiple users the same concrete task can still give a sense of where the problems lie. Some usability practitioners simply invite the user to explore the application or carry out a task that he or she defines. This may lead to data that is more relevant to actual user practice than assigning fixed tasks but the results are less guaranteed to be interpretable. In addition to examining the users' actions, practitioners

can gain additional insight into a system's usability by asking the users to *think aloud*, that is, to verbally report on their thinking as they interact with the system (or immediately afterwards). Think aloud protocols, can illuminate users' intentions in a way that mouse movements alone cannot. If protocols are systematically analyzed (coded and tallied) they can be a particularly detailed source of information but this process is often too labor intensive for the tight schedules that are typical of software development.

In the actual administration of the test, the tester may be in the testing room with the user or in a usually-adjacent observation room, communicating with the user via intercom. In either case, the tester plays a relatively passive role. The main role of the tester is to keep the user on track and solicit clarification of particular actions (e.g. "What were you thinking when you clicked 'SUBMIT'?"). It is common practice to ask users to react to design proposals ("What would you think if that was a set of radio buttons instead of a pull down menu?") but the point is not to collect the users' design ideas because users often have poor intuitions about the usability of particular features..

Laboratory requirements. In order to collect detailed usability data from test users, certain laboratory facilities, equipment and procedures are important. This section reviews typical requirements, based on our configuration and experience at the Bureau of Labor Statistics. Simpler configurations can be helpful as well; more complicated ones certainly exist but it's not clear that added complexity is a virtue for this type of facility.

- a) Space: a testing room and an observation room are the primary space requirements. They are typically separated by one way glass, allowing observers to see activity in the testing room. The two rooms are connected by an intercom so that the tester and users can speak to each other.
- b) Computing hardware: Whatever hardware is required to run the to-be-evaluated software needs to be situated in the testing room. It should be representative of what actual users will really interact with. So, typically, preparation for a test involves installing software on a dedicated computer. Since there do not seem to be standards for current voting hardware, it seems likely that the entire system (hardware and software) will need to be acquired for each test. Moreover, when specialized hardware (i.e. not PC based) is involved, the video capture (discussed in the next point) may have to be adapted to non-standard screen sizes and, potentially, non-standard video display. For example, with conventional personal computers – whether desktop or laptop models – it is possible to intercept and record the image displayed on the screen. It's not clear that this is possible for all voting systems.
- c) Video capture: The usability lab must have some way to record user actions for later review and analysis. User actions are typically captured on video (on either a VCR or DVD recorder). Often the user actions that lead to perceptible changes to the user interface are sent straight to the recording device via a device called a scan converter. This converts the image bound for the screen (e.g. a VGA image) into one that can be recorded (e.g. a VHS image). Scan conversion inevitably degrades the resolution of the image (VHS involves far fewer pixels than VGA) but seems preferable to pointing a camera at the screen. The image from aiming a

camera at the screen can be obscured by the user and usually is degraded by a continually rolling lines that result from asynchrony between the camera and screen. The degradation from the scan converter is tolerable if text on the screen is not too small and/or viewers are acquainted with the text. Ideally, the video sent to the monitor would be captured internally on the computer and stored in a standard digital video format (e.g. MPEG) but, to my knowledge, there still is commercially available method to do this.

In addition to the activity on the screen, there is other informative user behavior that should be captured with video cameras. In particular, an image of the users' hands while he or she uses the mouse (or other pointing device) and keyboard and possibly other artifacts (e.g. a sample ballot) can help explain and elaborate what appears on the screen. Suppose a voter intends to write in the name of a candidate and begins typing at the keyboard but no characters appear in the field for write-ins because she has failed to shift the system's focus to that field by touching it with the stylus. The user's typing does not appear on the screen so the attempted typing can only be recorded if the laboratory configuration includes a camera – usually overhead – to capture the user's hands. In addition, an image of the user's face can make his frustration crystal clear and particularly compelling to system developers who are generally skeptical about reports of usability problems. The face shot seems to be most interpretable when the camera is positioned directly in front of the user's face rather than to the side. Finally, we have found that a view from behind the user -- an over the shoulder view – can capture behaviors like pointing at screen objects which users sometimes do in order to explain their thinking. Because this kind of behavior does not have any consequence for the state of the user interface, it will be overlooked when considering the screen image alone.

Ideally, video cameras should be remotely controlled (pan, zoom, tilt) from the observation room. To the extent possible, they should be small and unobtrusive though users almost always become quickly engaged in the task and seem oblivious to cameras.

Additional video equipment that a usability lab should have includes a video mixer to combine multiple video images, a TV monitor in the observation room for observers to view all camera angles, video editing equipment to produce highlight tapes/disks composed of examples of user problems and a dedicated computer in the observation room with event logging software to code user actions for later tabulation.

Recruiting Users. There are several published arguments that a small number of users (e.g. 4-6) is enough to identify most unique usability problems. The rationale is that additional users primarily uncover problems that have already been identified (e.g. Hix & Hartson, 1993; Nielsen, J. & Levi, J., 1994). Nonetheless this is based on studies of relatively homogeneous user communities (unlike the users of voting software). The assumptions about the knowledge required for successful use of the interface must be

spelled out and, if successful use requires special knowledge, users with that knowledge must be recruited. If the user interface is intended to be usable by people without particular knowledge then users without that knowledge should be recruited. I return to this issue briefly at the end of the white paper.

User satisfaction questionnaires. While the lab measures primarily objective user behavior, some measure of the users' subjective experience is essential. Even if users are performing well, they will not use the system (or at least will not do so voluntarily) unless the experience is relatively pleasing and free of frustration. To assess subjective experience it is typical to administer a satisfaction questionnaire composed of both multiple choice and open ended questions immediately after the session. There are several commercially available questionnaires for which the items have been demonstrated to provide reliable measure but which are not tailored to any specific software product. As indicated above, more tailored questionnaires are often useful but they lack the psychometric stamp of approval.

Analyses. When possible, a comparison between multiple version of the interface provides context for the results, for example when one version leads to greater accuracy or satisfaction than the other. This is probably the most scientific approach to evaluating usability but is often hard to accommodate with real production deadlines. In any case, overall measures (e.g. proportion of tasks completed correctly and length of time per task) are often the most straightforward way to report and for readers to interpret the results. In general, high error rates, long completion times and low satisfaction suggest problems. These do always line up however. For examples, users may show no difference in satisfaction despite large differences in accuracy or various other combinations of values on these factors.

In contrast to such overall evaluations are event level analyses. These include counts of user actions like backtracking, changing previous actions, pulling down several menus in search of the intended one, inactivity accompanied by some suggestion of uncertainty, confusion or frustration such as gestures, verbal expressions and facial expressions, etc.. Users may eventually resolve problems of this sort and perform the overall task accurately but these low level problems slow them down and may degrade their experience. In practice, problems of this sort are often noted qualitatively by the tester.

Quantitative analyses – particularly at the event level – can be labor intensive and for many project staff they are less compelling than a list of frequent problems, especially when accompanied by video examples of real users having trouble. These make it clear to developers what they need to fix (although not necessarily how to do it.). The list is usually compiled on the basis on testers' notes, event logs, and accuracy/time measures. Despite the ease of interpreting a list of problems, quantitative summaries have more clout with technically oriented stake holders. Given the visibility of voting technology, quantitative justification for particular design decisions seems particularly important.

Usability and voting technology

Consider the following statements by manufactures of voting technology about their products. It seems unlikely their claims are based on the experience of actual voters interacting with the products. Real users are never as simple as designers assume.

*Since there are only three clear options, we can achieve error-free voting.
(www.vote-trakker.com)*

Whether or not the options are clear is an empirical question. As recent history makes evident, misunderstanding by even a handful of voters may be sufficient to change the outcome of the election.

Since you must choose between the three options of Contest Candidate, Write-in Candidate and No-Vote (Abstain), under and over votes are completely eliminated (www.vote-trakker.com)

This claim, even if true, is of little value if voters find the process to be unpleasant. The procedure for abstaining with paper ballots requires no action (not punching or not checking any option for a particular race) but the use of an Abstain button with this particular voting system requires voters to take an action (i.e. press the button). It's easy to imagine that not all voters using this product will make this mental translation. If such a voter believes she is abstaining and the system displays a message about undervoting, she is likely to be confused at best and, at worst, may give up.

*The voter completes a very simple and easy to follow voting process. When the voter casts the final vote by pressing the green button on the DVRM, the smart card is automatically deactivated. The DVRM stores all votes in an encrypted and password protected data bank only accessible by authorized election officials.
(Diversified Dynamics Inc., System 5 DVRS)*

The degree to which the voting process is simple and easy to follow is, again, an empirical question. One can argue that, if every vote counts and no one is to be disenfranchised, then 100% of test voters must follow the process without error or dissatisfaction.

Voter intent and ballot correctness are guaranteed prior to ballot casting. Over-votes cannot be accepted by the iVotronic and the voter is privately alerted of any undervotes during the final ballot review process

It's hard to get inside voters' heads but methods like thinking aloud in a usability test help expose their immediate goals and plans to achieve those goals. Surely, data from a usability test would help justify a claim like "voter intent ... [is] guaranteed."

A touch of the screen on the under-voted race will cause the AccuVote-TS to return the voter to the under-voted race within the ballot and allow the voter to complete the voting process. A voter can also step back and forth through the ballot, changing any selection until the ballot is "cast."

Diebold Election Systems

[<http://www.diebold.com/solutions/election/solutions.htm>]

This statement implies that it is easy for voters to navigate (“step back and forth”) through the ballot. But navigation is a notoriously thorny usability issue. For example, if scrolling is involved, voters can easily lose track of their location in the ballot which can be confusing.

Several of the systems whose promotional materials are excerpted above claim to prevent overvoting by blocking more than one vote per race but it remains to be seen how users react to this kind of system intervention when they do not believe they have voted more than once. For example, a stray finger movement may unintentionally register a vote for one candidate, voters may believe a ballot that spans more than one screen actually represents two races and may try to vote in each “race,” and voters may mistakenly believe that more than one candidate can be selected for a particular race. As a result, the content of the overvoting message is crucial. If it mystifies voters then they may fail to recover from their “error.” If it is unpleasant, they may be less likely to vote in a future election. Similarly, a system message to prevent undervoting, by warning the user they have not voted for any candidates for a particular race, may fluster users who actually intend to vote for no one.

Voter intent is a topic ripe for further investigation in the usability lab. Pressing a button – even a big button – does not guarantee that the user intended to trigger the particular action that the system interprets the button press to indicate. Here the notion of *slips* versus *mistakes* (Norman, 1981) is relevant. A slip is an unintended action like a stray finger movement grazing a touch screen while a mistake is an intended but erroneous action like deliberately pressing the “submit” button after choosing a candidate for the first race because the user believes that each vote must be submitted separately. In the case of button presses, the coordinates, duration and pressure of the action could be measured. A press near the edge of the button graphic may indicate less intention than a bull’s-eye; a very brief press may indicate less intention than a longer press though a very long press may indicate that the voter is resting the stylus or his finger on the button without intending to press it; a light press may indicate less intent than a firm one..

The final voting-specific usability issue I will raise here concerns the highly heterogeneous user base for voting systems. Many software products are intended for well defined and homogenous user communities and this drastically simplifies their design because certain levels of knowledge and experience can be assumed. However, this is not the case with voting where virtually every adult in the population needs to be able to successfully interact with the software. This poses special recruitment issues for

laboratory testing: Purposively sampling subgroups from within the population and testing system features believed to be potentially problematic for particular subgroups (e.g. font that is not large enough for the elderly) can help expose this kind of problem.

References

Bias, R.G. & Mayhew, D.J. (1994). *Cost-Justifying Usability*. Chestnut Hill, Mass: Academic Press.

Hix, D. & Hartson, H.R. (1993). *Developing User Interfaces : Ensuring Usability Through Product and Process*. NY: John Wiley & Sons

Nielsen, J. & Levi, J. (1994). Measuring usability: Preference vs. performance, *Communications of the ACM*, 37, 66-75.

Norman, D. (1981) Categorization of action slips. *Psychological Review*, 88 1-15.