Heuristic evaluation

This article is about usability evaluation. For a list of Heuristic analysis topics ranging from application of heuristics to antivirus software, see Heuristic analysis.

A **heuristic evaluation** is a usability inspection method for computer software that helps to identify usability problems in the user interface (UI) design. It specifically involves evaluators examining the interface and judging its compliance with recognized usability principles (the "heuristics"). These evaluation methods are now widely taught and practiced in the new media sector, where UIs are often designed in a short space of time on a budget that may restrict the amount of money available to provide for other types of interface testing.

Introduction

The main goal of heuristic evaluations is to identify any problems associated with the design of user interfaces. Usability consultant Jakob Nielsen developed this method on the basis of several years of experience in teaching and consulting about usability engineering.

Heuristic evaluations are one of the most informal methods[1] of usability inspection in the field of human-computer interaction. There are many sets of usability design heuristics; they are not mutually exclusive and cover many of the same aspects of user interface design.

Quite often, usability problems that are discovered are categorized—often on a numeric scale—according to their estimated impact on user performance or acceptance. Often the heuristic evaluation is conducted in the context of use cases (typical user tasks), to provide feedback to the developers on the extent to which the interface is likely to be compatible with the intended users' needs and preferences.

The simplicity of heuristic evaluation is beneficial at the early stages of design. This usability inspection method does not require user testing which can be burdensome due to the need for users, a place to test them and a payment for their time. Heuristic evaluation requires only one expert, reducing the complexity and expended time for evaluation. Most heuristic evaluations can be accomplished in a matter of days. The time required varies with the size of the artifact, its complexity, the purpose of the review, the nature of the usability issues that arise in the review, and the competence of the reviewers. Using heuristic evaluation prior to user testing will reduce the number and severity of design errors discovered by users. Although heuristic evaluation can uncover many major usability issues in a short period of time, a criticism that is often leveled is that results are highly influenced by the knowledge of the expert reviewer(s). This "one-sided" review repeatedly has different results than software performance testing, each type of testing uncovering a different set of problems.

Nielsen's heuristics

Jakob Nielsen's heuristics are probably the most-used usability heuristics for user interface design. Nielsen developed the heuristics based on work together with Rolf Molich in 1990.[1][2] The final set of heuristics that are still used today were released by Nielsen in 1994.[3] The heuristics as published in Nielsen's book *Usability Engineering* are as follows[4]

1. **Visibility of system status**:
   The system should always keep users informed about what is going on, through appropriate feedback within reasonable time..
2. **Match between system and the real world**:
   The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.
3. **User control and freedom**:
   Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo.
4. **Consistency and standards**:
   Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.
5. **Error prevention**:
   Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.
6. **Recognition rather than recall**:
   Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.
7. **Flexibility and efficiency of use**:
   Accelerators—unseen by the novice user—may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.
8. **Aesthetic and minimalist design**:
   Dialogues should not contain information which is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.
9. **Help users recognize, diagnose, and recover from errors**:
   Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.
10. **Help and documentation**:

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Gerhardt-Powals' cognitive engineering principles

Although Nielsen is considered the expert and field leader in heuristics, Jill Gerhardt-Powals also developed a set of cognitive principles for enhancing computer performance.[5] These heuristics, or principles, are similar to Nielsen's heuristics but take a more holistic approach to evaluation. Gerhardt Powals' principles[6] are listed below.

- **Automate unwanted workload**:
  - free cognitive resources for high-level tasks.
  - eliminate mental calculations, estimations, comparisons, and unnecessary thinking.

- **Reduce uncertainty**:
  - display data in a manner that is clear and obvious.

- **Fuse data**:
  - reduce cognitive load by bringing together lower level data into a higher-level summation.

- **Present new information with meaningful aids to interpretation**:
  - use a familiar framework, making it easier to absorb.
  - use everyday terms, metaphors, etc.

- **Use names that are conceptually related to function**:
  - Context-dependent.
  - Attempt to improve recall and recognition.
  - Group data in consistently meaningful ways to decrease search time.

- **Limit data-driven tasks**:
  - Reduce the time spent assimilating raw data.
  - Make appropriate use of color and graphics.

- **Include in the displays only that information needed by the user at a given time**.
- **Provide multiple coding of data when appropriate**.
- **Practice judicious redundancy**.

Weinschenk and Barker classification

Susan Weinschenk and Dean Barker created a categorization of heuristics and guidelines by several major providers into the following twenty types:[7]

1. User Control: heuristics that check whether the user has enough control of the interface.

2. Human Limitations: the design takes into account human limitations, cognitive and sensorial, to avoid overloading them.

3. Modal Integrity: the interface uses the most suitable modality for each task: auditory, visual, or motor/kinesthetic.

4. Accommodation: the design is adequate to fulfill the needs and behaviour of each targeted user group.

5. Linguistic Clarity: the language used to communicate is efficient and adequate to the audience.

6. Aesthetic Integrity: the design is visually attractive and tailored to appeal to the target population.

7. Simplicity: the design will not use unnecessary complexity.

8. Predictability: users will be able to form a mental model of how the system will behave in response to actions.

9. Interpretation: there are codified rules that try to guess the user intentions and anticipate the actions needed.

10. Accuracy: There are no errors, i.e. the result of user actions correspond to their goals.

11. Technical Clarity: the concepts represented in the interface have the highest possible correspondence to the domain they are modeling.

12. Flexibility: the design can be adjusted to the needs and behaviour of each particular user.

13. Fulfillment: the user experience is adequate.

14. Cultural Propriety: user's cultural and social expectations are met.

15. Suitable Tempo: the pace at which users works with the system is adequate.

16. Consistency: different parts of the system have the same style, so that there are no different ways to represent the same information or behavior.

17. User Support: the design will support learning and provide the required assistance to usage.

18. Precision: the steps and results of a task will be what the user wants.

19. Forgiveness: the user will be able to recover to an adequate state after an error.

20. Responsiveness: the interface provides enough feedback information about the system status and the task completion.

See also

- Usability inspection
- Progressive disclosure
- Cognitive bias
- Cognitive dimensions, a framework for evaluating the design of notations, user interfaces and programming languages

References

1. ^ ***a*** ***b*** Nielsen, J., and Molich, R. (1990). Heuristic evaluation of user interfaces, Proc. ACM CHI'90 Conf. (Seattle, WA, 1–5 April), 249-256
2. ^ Molich, R., and Nielsen, J. (1990). Improving a human-computer dialogue, Communications of the ACM 33, 3 (March), 338-348
3. ^ Nielsen, J. (1994). Heuristic evaluation. In Nielsen, J., and Mack, R.L. (Eds.), Usability Inspection Methods, John Wiley & Sons, New York, NY
4. ^ *Nielsen, Jakob (1994). Usability Engineering. San Diego: Academic Press. pp. 115–148. ISBN 0-12-518406-9.*
5. ^ *Gerhardt-Powals, Jill (1996). "Cognitive engineering principles for enhancing human - computer performance". International Journal of Human-Computer Interaction* ***8*** *(2): 189–211. doi:10.1080/10447319609526147.*
6. ^ Heuristic Evaluation - Usability Methods – What is a heuristic evaluation? Usability.gov
7. ^ *Jeff Sauro. "What's the difference between a Heuristic Evaluation and a Cognitive Walkthrough?". MeasuringUsability.com.*

Further reading

- Dix, A., Finlay, J., Abowd, G., D., & Beale, R. (2004). Human-computer interaction (3rd ed.). Harlow, England: Pearson Education Limited. p324
- Gerhardt-Powals, Jill (1996). Cognitive Engineering Principles for Enhancing Human-Computer Performance. "International Journal of Human-Computer Interaction", 8(2), 189-21
- Hvannberg, E., Law, E., & Lárusdóttir, M. (2007) "Heuristic Evaluation: Comparing Ways of Finding and Reporting Usability Problems", Interacting with Computers, 19 (2), 225-240
- Nielsen, J. and Mack, R.L. (eds) (1994). Usability Inspection Methods, John Wiley & Sons Inc

External links

- Jakob Nielsen's introduction to Heuristic Evaluation - Including fundamental points, methodologies and benefits.

- Alternate First Principles (Tognazzini) - Including Jakob Nielsen's ten rules of thumb
- Heuristic Evaluation at Usability.gov
- Heuristic Evaluation in the RKBExplorer
- Remote (online) Heuristic Evaluation Tool at usabiliTEST.com.