CHAPTER 1

Heuristic Evaluation

User Interface Inspection Methods. DOI: http://dx.doi.org/10.1016/B978-0-12-410391-7.00001-4

Alternate Names: Expert review, heuristic inspection, usability inspection, peer review, user interface inspection.

Related Methods: Cognitive walkthrough, expert review, formal usability inspection, perspective-based user interface inspection, pluralistic walkthrough.

OVERVIEW OF HEURISTIC EVALUATION

A heuristic evaluation is a type of user interface (UI) or usability inspection where an individual, or a team of individuals, evaluates a specification, prototype, or product against a brief list of succinct usability or user experience (UX) principles or areas of concern (Nielsen, 1993; Nielsen & Molich, 1990). The heuristic evaluation method is one of the most common methods in user-centered design (UCD) for identifying usability problems (Rosenbaum, Rohn, & Humburg, 2000), although in some cases, what people refer to as a heuristic evaluation might be better categorized as an expert review (Chapter 2) because heuristics were mixed with additional principles and personal beliefs and knowledge about usability.

A *heuristic* is a commonsense rule or a simplified principle. A list of heuristics is meant as an aid or mnemonic device for the evaluators. Table 1.1 is a list of heuristics from Nielsen (1994a) that you might give to your team of evaluators to remind them about potential problem areas.

There are several general approaches for conducting a heuristic evaluation:

- **Object-based**. In an object-based heuristic evaluation, evaluators are asked to examine particular UI objects for problems related to the heuristics. These objects can include mobile screens, hardware

Table 1.1 A Set of Heuristics from Nielsen (1994a)

Example List of Heuristics
1. Visibility of system status
2. Match between system and the real world
3. User control and freedom
4. Consistency and standards
5. Error prevention
6. Recognition rather than recall
7. Flexibility and efficiency of use
8. Aesthetic and minimalist design
9. Help users recognize, diagnose, and recover from errors
10. Help and documentation

control panels, web pages, windows, dialog boxes, menus, controls (e.g., radio buttons, push buttons, and text fields), error messages, and keyboard assignments.

- **Task-based**. In the task-based approach, evaluators are given heuristics and a set of tasks to work through and are asked to report on problems related to heuristics that occur as they perform or simulate the tasks.
- **An object–task hybrid**. A hybrid approach combines the object and task approaches. Evaluators first work through a set of tasks looking for issues related to heuristics and then evaluate designated UI objects against the same heuristics. The hybrid approach is similar to the heuristic walkthrough (Sears, 1997), which is described later in this book.

In task-based or hybrid approaches, the choice of tasks for the team of evaluators is critical. Questions to consider when choosing tasks include the following:

- **Is the task realistic**? Simplistic tasks might not reveal serious problems.
- **What is the frequency of the task**? The frequency of the task might determine whether something is a problem or not. Consider a complex program that you use once a year (e.g., US tax programs). A program intended to be used once a year might require high initial learning support, much feedback, and repeated success messages—all features intended to support the infrequent user. These same features might be considered problems for the daily user of the same program (e.g., a tax accountant or financial advisor) who was interested in efficiency and doesn't want constant, irritating feedback messages.

- **What are the consequences of the task**? Will an error during a task result in a minor or major loss of data? Will someone die if there is task failure? If you are working on medical monitoring systems, the consequences of missed problems could be disastrous.
- **Are the data used in the task realistic**? We often use simple samples of data for usability evaluations because it is convenient, but you might reveal more problems with "dirty data."
- **Are you using data at the right scale**? Some tasks are easy with limited data sets (e.g., 100 or 1000 items) but very hard when tens of thousands or millions of items are involved. It is convenient to use small samples for task-based evaluations, but those small samples of test data may hide significant problems.

Multiple evaluators are recommended for heuristic evaluations, because different people who evaluate the same UI often identify quite different problems (Hertzum, Jacobsen, & Molich, 2002; Molich & Dumas, 2008; Nielsen, 1993) and also vary considerably in their ratings of the severity of identical problems (Molich, 2011).

The Evaluator Effect in Usability Evaluation

The common finding that people who evaluate the usability of the same product report different sets of problems is called the "evaluator effect" (Hertzum & Jacobsen, 2001; Jacobsen, Hertzum, & John, 1998a,b). The effect can be seen in both testing and inspection studies. There are many potential causes for this effect including different backgrounds, different levels of expertise, the quality of the instructions for conducting an evaluation, knowledge of heuristics, knowledge of the tasks and environment, knowledge of the user, and the sheer number of problems that a complex system (e.g., creation-oriented applications like PhotoShop and AutoCAD), with many ways to use features and complete tasks, can present to users (Akers, Jeffries, Simpson, & Winograd, 2012). Knowing that evaluators will find different problems, from the practical point of view, can be dealt with by:

- Using multiple evaluators with both UX and domain knowledge.
- Training evaluators on the method and materials used (checklists, heuristics, tasks, etc.). Providing examples of violations of heuristics and training on severity scales can improve the quality of inspections and walkthroughs.
- Providing realistic scenarios, background on the users, and their work or play environments.

- Providing a common form for reporting results and training people on how to report problems.
- Providing evaluators with the UX dimensions (e.g., learnability, memorability efficiency, error prevention, and aesthetics) that are most critical to users.
- Considering how something might be a problem to a novice and a delighter to an expert.

This book discusses the strengths and weaknesses of each approach and provides tips from academic and practical perspectives on how to make inspections and walkthroughs more effective.

During the heuristic evaluation, evaluators can write down problems as they work independently, or they can think aloud while a colleague takes notes about the problems encountered during the evaluation. The results of all the evaluations can then be aggregated into a composite list of usability problems or issues and often prioritized based on severity, predicted frequency of occurrence, or other important dimensions (Yehuda & McGinn, 2007).

The literature on heuristic evaluation suggests that sessions should last one to two hours, not including the time required for training the people who will be the evaluators. This is a conservative estimate. Depending on the complexity of the product and the scope of the reviews, heuristic evaluations by a dedicated team might take days, with some invited participants focusing on particular areas for a few hours. In addition, the amount of time required to aggregate the results can be considerable depending on the number of evaluators, the method for reporting the problems (especially the granularity of the problem), and the complexity of the product.

Setting the Scope of Inspections

If you are asked to conduct a heuristic inspection by member of a product team or a client, make sure that you clarify the scope of the inspection. Open-ended inspections with no clear review boundaries can consume considerable time. Be explicit about what pages, screens, features, and UI objects are the focus of the inspection.

Heuristic evaluation is a popular method and one that is taught in many graduate human–computer interaction (HCI) programs, but also one of the more controversial for several reasons:

- **In some environments, the politics of inspections can be more challenging than the politics of actual testing**. Inspection results may be perceived as "just opinions," even though the results of inspections are of similar quality to usability testing (Molich, 2013).
- **Heuristics are often very general, and evaluators may have different conceptions of what the heuristics mean**. One early heuristic (Nielsen, 1993) is just "Consistency and Standards," which is so broad as to be nearly useless. Consistency, for example, is a complex issue and a simple heuristic does not capture this complexity (Wilson, 2009).
- **Evaluators are asked to use a prescribed set of heuristics as a guide, but most evaluators also apply subjective judgment and other principles** as the basis for reporting problems that often make the heuristic evaluation more of an individual expert review (see Chapter 2).
- **Evaluators report problems at different levels of granularity**; sometimes evaluators report things at a categorical level ("error messages are cryptic and lacking helpful information") and sometimes at an atomic level (Error message 132 has cryptic text that says "sysadm overflow: tetricular overflow—file is about to be conjugated").
- **It is easy to catch surface issues**—the poor text in a message, misalignment of controls, and superfluous text—but it is much harder to capture larger workflow issues. Sometimes the problems found in heuristic evaluation are "surface problems"—misalignments, ugly icons, and labels that are not quite clear. Those can all affect the UX, but deeper, task-related problems might be more critical issues for users.
- **Heuristic evaluation is prone to "false positives," reported problems that are, in actual use, not problems**. False positive can emerge because of incomplete knowledge of how people will use a product, evaluators' underestimation of user skills and adaptability, lack of understanding of usability and design principles, and flawed judgments about how an alternative design might perform in a particular context (Gilbert & Woolrych, 2001).

These issues are discussed later in the chapter with some recommendations about how to deal with their related controversies and minimize their impact.

WHEN SHOULD YOU USE HEURISTIC EVALUATION?

The primary goal of a heuristic evaluation is to reveal as many usability or design problems as possible at a relatively low cost. Heuristic evaluation, after all, was designed to be a discount usability method (relative to testing in a usability laboratory). A secondary (though very important) goal of the heuristic evaluation is to train members of the product team to recognize potential usability problems so they can be eliminated earlier in the design process.

Heuristic evaluations are a good fit in the following situations:

- You have limited (or no) access to users.
- You have an appropriate mix of design and domain expertise among potential evaluators.
- You need to produce an extremely fast review and do not have time to recruit participants and set up a full-fledged laboratory study.
- Your evaluators are dispersed around the world—a common situation.
- You are looking for breadth in your review. While usability testing is generally considered the best approach for finding usability problems in a product, most usability testing is designed to cover only small portions of products and services. Heuristic evaluation can provide additional breadth and complement other assessment techniques.
- Your clients have come to trust your judgment and, for many issues, will accept your recommendations without requiring you to conduct user testing or other more expensive evaluation methods. Of course, this carries some significant risk if a trusted usability practitioner fails to catch some very serious problems that result in schedule slips and lost revenue, but the same thing could happen in a usability test that doesn't include tasks that expose severe problems.
- Your project lacks significant funding for usability testing.

Heuristic evaluations can be conducted at any phase of the development cycle after problem definition (Table 1.2) where there is some representation of the UI (UI specification, functional specification, detailed storyboards, paper prototypes, working prototypes, or working product). Heuristic evaluations can be conducted iteratively from requirements definition to implementation to find and filter out usability issues and minimize rework by development.

Table 1.2 Phases of Development When Heuristic Evaluations Are Useful

	✓	✓	✓	✓
Problem Definition	Requirements	Conceptual Design	Detailed Design	Implementation

Table 1.3 Relative Effort and Resources Required for Heuristic Evaluation—More Colored Bars Mean More Resources

Overall Effort Required	Time for Planning and Conducting	Skills and Experience	Supplies and Equipment	Time for Data Analysis
🟦🟦⬜⬜⬜	🟦🟦⬜⬜⬜	🟦🟦⬜⬜⬜	🟦⬜⬜⬜⬜	🟦🟦⬜⬜⬜

Table 1.3 illustrates the relative effort required, on average, to develop and use heuristic evaluation. Many heuristic evaluations require only limited resources; however, if your evaluation is used for high-risk situations, the resources (e.g., more evaluators and development of custom heuristics) required may increase considerably.

STRENGTHS

Heuristic evaluation has the following strengths:

- **The heuristic evaluation method is simple to explain (although the word "heuristic" is not)**. At its most basic, you hand people a list of heuristics with some explanation and examples, provide them with a representation of the UI to review and ask them to list usability problems using the heuristics as a guide.
- **Heuristic evaluation is relatively fast if your focus is on a reasonable scope of features**. Heuristic evaluation can provide useful data relatively quickly, without the expense or effort associated with recruiting users.
- **Heuristic evaluations are similar to software code inspections**, and this similarity may make heuristic evaluations easier for product teams to accept than other usability evaluation methods.
- **Heuristic evaluations require no special resources** such as a usability laboratory (although for early prototypes of the product, you may need to work with a "test system") and can be used across a wide variety of products at different stages of development.
- **Heuristic evaluations increase awareness of common usability problems and serve as a method for training the product team about what**

aspects of design can lead to usability problems. One hidden strength of heuristic evaluation and the other methods in this book is that over the course of a year, colleagues who are involved in the evaluations will start to recognize design issues very early and eliminate at least some categories of problems.

WEAKNESSES

Heuristic evaluation has the following weaknesses:

- **Different evaluators often find different problems for the same product**. This "evaluator effect" (Jacobsen et al., 1998a,b) has implications for deciding what changes should be made to a design. What do you do if five evaluators each come up with quite different sets of problems (Kotval, Coyle, Santos, Vaughn, & Iden, 2007)? One tip here is to have a group meeting of all the evaluators where you walk through the problems, one by one, and discuss their validity and severity. The group discussion can help to determine which problems are "real" and which are "false positives" and to develop a consensus on the relative severity of different problems. Group reviews can also yield additional (missing) problems through the group interaction.
- **The heuristic evaluation method is based on finding usability problems**, but there is debate about what constitutes a problem and whether heuristic reviews are good at finding "real problems" (see a discussion of this in the "Major Issues in the Use of the Heuristic Evaluation" section). A problem for a beginner might be perceived as a positive feature for an expert and vice versa.
- **Heuristic reviews may not scale well for complex interfaces such as architectural design tools or Adobe Photoshop (Slavkovic & Cross, 1999)**. In complex interfaces, a small number of evaluators may find only a small percentage of the problems in an interface and may miss some serious problems. If you are evaluating complex tools, you may need to use multiple methods and combine the results across those methods.
- **Evaluators may not be the actual users of the system**. To strengthen the results of a heuristic evaluation, it is useful to involve domain specialists or to conduct several interviews with actual users to understand more about how they use the product.

- **In some situations, the evaluators have a vested interest in the product, which might blind them to some problems**. It is often useful to involve one or two outsiders who do not have a direct vested interest in the product to serve on the evaluation team.
- **Evaluators may report problems at different levels of granularity**. For example, one evaluator may list a global problem of "bad error messages" while another evaluator lists separate problems for each error message encountered. The instructions and training for a heuristic review should discuss the appropriate level of granularity. The facilitator of a heuristic evaluation will invariably have to extract important high-level issues from sets of specific problems.
- **Lack of clear rules for assigning severity judgments may yield major differences**; one evaluator says "minor" problem while others say "moderate" or "serious" problems. This author has examples of heuristic evaluations where different reviewers listed the same observation as both a problem and a "positive design feature."
- **Heuristic evaluation depends strongly on the quality and experience of the evaluators**. The value of conducting a heuristic evaluation with five novices who have limited knowledge of the domain and users may be questionable. The results of heuristic evaluation can match those of usability testing when your evaluators have ten years or more of experience in the usability domain (Molich, 2013).
- **Heuristic evaluation does not address solutions to problems**. Solutions are generally recommended by the facilitator or arrived at by discussion with the product team.

Tips on Providing Solutions with the Results of a Heuristic Evaluation

1. Examine your problem set for global problems and propose a solution that will work for many individual instances of the same global problem.
2. If you found a number of comments about the organization of controls and information on a page, window, or dialog box, consider sketching a few rough prototypes of an improved design. Drawing a few rough solution sketches (rather than a single solution) gives developers some choice and increases the likelihood that they will take your suggestions seriously.
3. Provide a brief rationale for your recommendations on how to solve the problems. For example, you might explain that providing

a shortcut will improve productivity because the user does this task many times a day.

4. Make sure that your proposed solutions are consistent among themselves. Check for common patterns in the product or service and if the problem occurs in multiple places, consider whether a single solution will work.

- **Heuristic evaluation may not reveal problems that involve complex interactions by multiple users (e.g., in a collaborative work environment)**.

Evaluations of Complex System May Require More Evaluators and Multiple Sets of Heuristics

Jaferian, Hawkey, Sotirakopoulos, Velez-Rohas, and Beznosov (2011) conducted a study of a very complex and collaborative environment—IT security management (ITSM)—comparing the number and severity of problems found using Nielsen's (1994a) heuristics versus a specialized set of ITSM heuristics. The result was that the two groups (ITSM heuristics versus Nielsen heuristics) of evaluators found different types of problems. The evaluators using the ITSM heuristics found more severe problems than the Nielsen heuristics group. The authors suggested that for complex systems (1) more evaluators are necessary when systems are complex, collaborative, and run by people with different risk tolerance, and that (2) it would be worthwhile to use both the Nielsen and domain-focused heuristics.

- **Product team members may feel that the heuristic evaluation is like a performance appraisal of their design skills and be nervous about this**. It is critical to let people know that it's good to find problems and that the results won't be used in any way as a measure of individual performance or skill. One way to mitigate the impact of a list of problems on colleagues or clients is to provide key positive aspects of the evaluation. The issue of reporting positive findings is discussed later in this chapter.

WHAT RESOURCES DO YOU NEED TO CONDUCT A HEURISTIC EVALUATION?

This section provides a brief description of the basic resources needed to conduct a heuristic evaluation.

Personnel, Participants, and Training

You need a facilitator who coordinates the activities of the evaluation team. The facilitator prepares the materials, conducts training (on the heuristic evaluation process, the heuristics chosen for the evaluation, and the features to be reviewed), arranges a representation of the UI to be available, collects the evaluations, compiles and analyzes the results, organizes a group review if necessary, prepares a report, and develops a heuristic evaluation infrastructure to track the results and examine trends and patterns in the data.

Facilitators need to be trained in the following:

- How to integrate this method with other usability evaluation methods
- Who to choose for the evaluation team
- How to integrate the problems from multiple reviewers
- The strengths and weaknesses of the method
- Methods for determining how serious the problems are
- Managing the different perspectives that various evaluators bring to the table, especially when the evaluators are from different groups, each having their own interest in the design.

A heuristic evaluation generally requires a team ranging from two to ten people (Hwang & Salvendy, 2010) with a mixture of usability, product, domain, and work experience. The size of the team will vary depending on the levels of experience of the evaluator, the scope of the evaluation, and the complexity of the software, tasks, and environment. Heuristic evaluation teams can benefit from some reviewers who are outside the product team to spot problems that might go unnoticed by those who are experienced with the product.

Depending on the size of your evaluation team, you may need assistance in compiling and debugging the results (determining whether different narrative accounts of the problems are in fact the same problem or different problems). If your heuristic evaluation procedure calls for think-aloud sessions where the evaluator reviews the product and a note taker records the problems, you will need note takers (preferably with some usability and product experience so they provide accurate descriptions of problems and heuristics).

Members of the heuristic evaluation team need to be trained in the following:

- How to go about the evaluation (using scenarios, screen shots, etc.)
- Descriptions and explanations of the heuristics used in a particular study
- How to assign heuristics to problems
- How to record problems using standard terminology (set in advance) for the same issues
- Severity assessments.

Hardware and Software

Besides the obvious requirement to provide the evaluators with the specification, product, or service they are evaluating, you also need to consider the following:

- Software forms for entering problems (Microsoft Word documents, spreadsheets, or forms that put problems directly into a database).
- A database for tracking problems, solutions, fixes, and whether subsequent reviews or testing indicate whether the problem is actually fixed. As noted earlier, the number of problems found is a common metric in research, but in the world of the practitioner, the number of problems fixed and subsequent verification that the fix yielded an improvement are the more critical metrics.

Documents and Materials

The following documents and materials are necessary for heuristic evaluation:

- A list of heuristics with brief explanations and examples
- Training materials (this is optional, but useful especially in a corporate environment where there will be repeated heuristic evaluations)
- Forms for entering problems (these can be online)
- A list of task scenarios that you want reviewers to follow and/or the particular parts of the UI that will be reviewed.

Too Much Detail in Task Scenarios Can Backfire!

Keep in mind that the level of detail you provide in your task scenarios for the evaluators may affect what problems emerge. Providing great detail in your task scenarios tends to limit the number of paths the evaluator may take while working on a task, thus reducing the potential breadth of problems that will emerge.

- A catalog of screen shots, wireframes, prototypes, or the UI/functional specification for the product (if it isn't yet working). Even if you have a working prototype or product, a catalog of screen shots is a useful way to provide context for the evaluators. This author has found it quite useful to have a screen shot at the top of a page followed by the problem entry form below.
- Report templates suitable to both executive audiences and to the members of the product team who are responsible for making changes based on the heuristic evaluation.
- A set of ground rules for the heuristic evaluation and the meeting of the evaluation team to review the results of the heuristics evaluation.

PROCEDURES AND PRACTICAL ADVICE ON THE HEURISTIC EVALUATION METHOD

The procedures for conducting a heuristic can vary widely in formality. The least formal approach is to gather together several colleagues, give them a list of heuristics and a representation of the product (e.g., UI specification, prototype, high-fidelity working model), and then ask them to fill out a defect or problem form that describes potential problems and the heuristics associated with the problems. At the other end of the spectrum is a more formalized procedure. Evaluators in this formalized procedure have the following characteristics:

1. **Evaluators use existing heuristic tailored to the system being evaluated or create context-specific heuristics**.

Developing Tailored Heuristics

There are two basic ways to create heuristics that are tailored to your specific product. You can take a large number of problems from the type of system you are developing (from support databases, previous design reviews, usability testing, field studies), classify them into common themes, and then refine those themes into heuristics. A second approach is to extract heuristics from a detailed literature review. For example, Tsui, Abu-Zahra, Casipe, M'Sadoques, and Drury (2010) developed a set of heuristics for assistive robotics by reviewing literature on accessibility, social robotics, cognitive psychology, perceptual psychology, and motor interactions with robotics. Desurvire and Wiberg (2008) examined the HCI, social learning, and self-efficacy research literature to develop a set of game approachability principles (GAP) that could be used for heuristic evaluation of beginning levels of game design.

---

2. Evaluators are trained on heuristics and the heuristic evaluation process to assure mutual understanding.
3. Evaluators are required to evaluate explicit features and apply specific usability criteria to each feature (Yehuda & McGinn, 2007).
4. Evaluators meet as a group to discuss their individual lists and combine the problems into a single list with agreed-on severity levels.
5. Evaluators develop one or more solutions for each problem or problem category.
6. Evaluators determine the effectiveness of the heuristic evaluation by tracking what problems were fixed, what solutions were implemented (compared to those suggested in the report), how much effort was expended on the change, and if there are meta-problems that should be addressed.

The next section presents general procedures for a heuristic evaluation. You will need to adapt these procedures to your particular organizational environment, development process, staff, and resources.

Planning a Heuristic Evaluation Session

Heuristic evaluations are considered informal, and one of the oft-cited advantages of the method is that "it does not require advance planning" (Nielsen & Molich, 1990, p. 255). On very small projects where the usability practitioners are well integrated into the process, this assertion can be true. But in many product development environments, advanced and detailed planning (especially the first time) is required. Following are the basic planning steps:

1. **Choose a team of evaluators**. The best evaluators are those with both domain and usability experience (Desurvire, 1994; Nielsen, 1992). For example, if you were evaluating a new travel site, you might seek out professional usability practitioners with travel software experience. If the product is complex, and the evaluators do not have much familiarity with the product, consider having one or

more domain specialists work with the evaluation team. When you consider the size and staffing of your heuristic review team, consider the following issues:

- Heuristic evaluations should generally not depend on a single evaluator. An evaluation team of three to five people is reasonable based on some modeling data and a cost–benefit analysis; however, the actual size of an evaluation team will depend on the following:
  - The heuristic and domain skills of your evaluators (Karoulis & Pombortsis, 2004). Kirmani and Rajasekaran (2007) provide a preliminary approach for assessing heuristic evaluation skills by comparing evaluators on the number of problems they find and the severity of the problems.
  - The priority associated with the product.
  - The complexity of the interface.
  - Your organization's tolerance of risk.
  - The breadth and depth of features that you are expected to cover.
  - Whether the system is mission-critical for your customers.
  - Whether the system presents safety or health hazards.

  For very large and complex systems (which include many websites, customer relationship management (CRM) software, mission-critical, and other high-risk applications), you may very well need more than three to five evaluators.
- **How much effort will it take to compile the individual lists into an aggregate list and prepare a report for your client**? Pulling together the evaluations from a large team with different views of problems, severities, and heuristics can be a daunting task. A short training session and the use of common (and simple) forms for reporting problems that allow easy merging of results can reduce the compilation effort.
- **What is the best mix of evaluators**? Double experts (those who have both domain and usability or HCI expertise) should be on evaluation teams if possible (Kirmani & Rajasekaran, 2007; Stone, Jarrett, Woodroffe, & Minocha, 2005). If there are few or no double experts, then you need to try and include both usability experts and domain experts. Other candidates for the evaluation team include quality engineers, technical writers, technical support specialists, technical analysts, information architects, and training specialists. You should also include a few evaluators

from different projects who can look at the product with fresh eyes to avoid the problem of "familiarity blindness." For products that will have broad consumer use, you might consider inviting some nonexperts as well as some edge users who might stretch the limits of the product and reveal usability problems that would otherwise go unnoticed. Cockton Woolrych, Hornbæk, and Frøkjær (2012) recommend that evaluation teams include people who have a good mix of the following distributed cognitive resources (DCRs) to find the widest range of problems:

- **User knowledge** (knowledge of user skills, abilities, training, and work environments)
- **Task knowledge** (detailed knowledge of the task and more importantly, how users perform in both individual and collective tasks)
- **Domain knowledge** (knowledge of the target domain—finance, engineering, games)
- **Design knowledge** (knowledge and experience with UI design, interaction design, and visual design)
- **Interaction knowledge** (how do people really work with a system, which might not be how product teams think they use the system)
- **Technical knowledge** (e.g., knowledge of browsers, cloud computing, and iOS)
- **Product knowledge** (specific knowledge of the target product, its features, and capabilities).

2. **Decide which heuristics are most useful for your evaluation**. Numerous sets of general heuristics are available. Table 1.4 provides four examples of different sets of heuristics.
   The major issues around the choice of heuristics include the following:
   - **Relevance**. Are the heuristics relevant to the domain and product? Some general heuristics like those in Table 1.4 are general enough to be relevant to many types of computing systems. However, if you are evaluating a call center application where extreme efficiency is a key attribute, you may need to include some domain-specific heuristics that are relevant to the call center environment and focus on high efficiency. As discussed earlier in this chapter, a combination of general heuristics and domain heuristics might be the best approach for more complex products and services.

Table 1.4 Examples of Sets of Heuristics

Nielsen Heuristics (Nielsen, 1994b)	Groupware Heuristics (Baker, Greenberg, & Gutwin, 2002)
1. Visibility of system status	1. Provide the means for intentional and appropriate verbal communication
2. Match between system and the real world	2. Provide the means for intentional and appropriate gestural communication
3. User control and freedom	3. Provide consequential communication of an individual's embodiment
4. Consistency and standards	4. Provide consequential communication of shared artifacts (i.e., artifact feedthrough)
5. Error prevention	5. Provide protection
6. Recognition rather than recall	6. Manage the transitions between tightly coupled and loosely coupled collaboration
7. Flexibility and efficiency of use	7. Support people with the coordination of their actions
8. Aesthetic and minimalist design	8. Facilitate finding collaborators and establishing contact
9. Help users recognize, diagnose, and recover from errors	
10. Help and documentation	
Research-Based Heuristics (Gerhardt-Powals, 1996)	General Heuristics (Weinschenk & Barker, 2000)
1. Automate unwanted workload	1. User control
2. Reduce uncertainty	2. Human limitations
3. Fuse data	3. Modal integrity
4. Present new information with meaningful aids to interpretation—use a familiar framework, making it easier to absorb	4. Accommodation
	5. Linguistic clarity
	6. Aesthetic integrity
5. Use names that are conceptually related to function	7. Simplicity
	8. Predictability
6. Group data in consistently meaningful ways to decrease search time	9. Interpretation
	10. Accuracy
7. Limit data-driven tasks	11. Technical clarity
8. Include in the displays only that information needed by the user at a given time	12. Flexibility
	13. Fulfillment
9. Provide multiple coding of data when appropriate	14. Cultural propriety
	15. Suitable tempo
10. Practice judicious redundancy (to resolve the possible conflict between heuristics 6 and 8)	16. Consistency
	17. User support
	18. Precision
	19. Forgiveness
	20. Responsiveness

- **Understandability**. Will the heuristics be understood by all members of the analysis team? One of the issues with the research-based guidelines (see Table 1.4) proposed by Gerhardt-Powals (1996) is that the evaluators have to be quite familiar with cognitive psychology and human factors research and principles to understand just what it means to "Limit data-driven tasks" and "fuse data." Even simple-sounding heuristics such as "chunking" can be misunderstood. There is also the issue of consistent understanding of the heuristics. Do you evaluators have a reasonably consistent understanding of heuristics such as "consistency" and "prevent errors"? There are many types of consistency and many types of error prevention. One way to improve understandability is to include subcategories under each heuristic such as:

  Consistency

  - Consistency with the way people work (Grudin, 1989; Wilson, 2009)
  - Terminology/language consistency
  - Visual consistency
  - Interaction consistency
  - Metaphorical consistency
  - Layout consistency
  - Consistency with other similar features in the existing product
  - Error prevention consistency
  - Consistency with operating system (OS) guidelines.

- **Use as memory aids**. Are the heuristics good memory aids for the many detailed guidelines they are meant to represent? For example, does the heuristic "error prevention" prompt the novice or expert to consider the hundreds of guidelines regarding good labeling, input format hints, the use of abbreviations, explicit constraints on the allowable range of values, and other techniques or principles for actually preventing errors?
- **Validity**. Is there proof that applying a particular set of heuristics will lead to better products? Bailey (1999) argues for the use of research-based heuristics where the literature shows that particular methods have an actual impact on performance.

3. **Develop an infrastructure for collecting, organizing, tracking, and reporting on the results of the heuristic evaluation**. The infrastructure can include the following:
   - Paper or online forms for collecting problems. These forms can include the following:
     - A brief description of the problem
     - Where the problem was found
     - The number of users who might experience the problem
     - The frequency/persistence of the problem for various classes of users
     - The severity of the problem
     - Whether the problem was local (found in only one place in the UI) or global (found in multiple places in the UI) (Dumas & Redish, pp. 322–323) (Table 1.5).

Table 1.5 Template for Collecting Problems in a Heuristic Evaluation

Problem/Problem Location	Heuristic	Severity	Frequency/ Persistence	Extent (Local versus Global)	Rationale/ Notes

- A training package where you explain, for example, what heuristics such as "match between system and the real world" really mean.
- One or more report formats geared to the needs of your clients. The reports are used to communicate the results of the evaluation, suggest possible solutions, and provide metrics for teams and managers.
- A database or some method for aggregating, organizing, and tracking problems.
- A method for tracking what problems will be dealt with by development or another group (e.g., some problems may be the responsibility of the documentation team) and some measure of the actual number of problems from the evaluation that were fixed.
- An explicit description of how to assess the severity of usability problems. This description should include definitions of severity levels, examples of problems for each severity level, and a discussion of why they received a particular severity rating.

4. **Conduct a short training session with potential evaluators who have not been part of earlier evaluations**. This step is optional, but from this author's experience, it's an activity that will increase the quality of reviews within an organization. Training can cover the heuristic evaluation process, the heuristics that will be used, the severity scale (if applicable), the reporting form, and the level of detail expected when describing usability problems. If your evaluators are new to the method, consider conducting a practice heuristic evaluation and discussing the results.
5. **Provide the evaluators with some context to help them understand the business objectives, the personas or user profiles, the user goals and tasks, and the environments where the product will be used**. The amount of background that you provide might vary according to

complexity. If you are evaluating a walk-up-and-use interface such as a mall kiosk, you might simply tell your evaluators the purpose of the system and not much else. If your product is complex, such as a financial trading system or big data visualization tool, the evaluators will need additional details (Lauesen, 2005).

6. **Provide the team with an overview of the prototype or working system that they will be evaluating**. This overview includes the following:
   - How to get access to the system
   - What the product is used for
   - The major features
   - Any limitations that they need to know about
   - Whether anything the evaluator does could cause any damage to the product that is not recoverable
   - A description of the sample data and some background on how realistic those data are
   - Who to call for help (prototype systems can often be temperamental, and reliable support can save time and reduce frustration).
7. **Choose which approach to your heuristic evaluation is most appropriate given the state of the product**. The basic approaches are listed here:
   - Create a set of important task scenarios, and ask your evaluators to work through the scenarios, noting any problems that emerge.
   - Give your evaluators a set of goals, and ask them to use the product to achieve those goals. For example, if you were testing a working prototype of an e-commerce website, you might ask each person on the team to use the prototype "buy something that they need in the next month" and record problems as they try to make their purchases.
   - Ask the evaluators to review specific UI objects (pages in a website, windows and dialog boxes, menus, error messages) and record violations of heuristics.
   - Ask the evaluators to evaluate a product based on a combination of task scenarios, goals, and specific UI objects and then report problems that emerge from this combination of approaches. This is the most powerful approach, but it takes more planning and effort.

Conducting a Heuristic Evaluation

Follow these steps to conduct a heuristic evaluation:

1. **Orient the evaluators**. Provide the team with the materials required for the evaluation of the particular representation of the UI that is

available (a catalog of screen shots, a paper prototype, a set of wire-frames with clickable links, a working prototype, or even a competitive product). Review the procedures with the entire group if possible, and field any questions that the evaluation team might have. Go over the schedule for the evaluation. Provide any passwords and access information that people didn't get in the training and overview session.

Provide an Exemplary Sample of a Problem Report

A best practice for heuristic evaluation or any other usability evaluation method is to provide an exemplary problem report that provides some clues to new evaluators about what makes a "great problem report." This exemplar can provide examples of the amount of detail needed and what information is critical. For example, it is important in a completed problem reporting form to note exactly where the problem occurs—which screen, dialog, page, window, or application.

2. **Ask the evaluators to conduct individual evaluations of the UI**. Because the evaluations are generally done individually, there is no way to strictly control how each person goes about the review, but you might suggest a particular review sequence such as the following:
   a. Familiarize yourself with any background information (goals, personas, environment descriptions) on the product.
   b. Walk through the task scenarios.
   c. Review any additional parts of the product that were not part of the task scenarios.
   d. If you have questions about the review process, contact the leader of the evaluation. If you have technical or domain questions, contact the designated technical expert.
3. **Advise your evaluators to list problems separately, even if the same problem recurs in different places in the UI**. It is important to know if problems are pervasive (global) or isolated (local). If you find the same problem in many places (e.g., a bad search UI in different areas of a product or bad error messages that occur in editable data tables), then this might alert you to systemic problems that require an architectural change.
4. **If your heuristic evaluation will last over several days, consider reconvening your evaluation team at the end of the first day of review so that issues about the procedure or product can be discussed and**

ironed out before they have gone too far in the evaluation. If a group meeting isn't possible, you might contact your evaluators and ask if they have any questions or issues.

5. **Collect the heuristic evaluation forms containing the list of problems found by each evaluator**. Let the evaluators know that you may call them if you have any questions.

After the Heuristic Evaluations by Individuals

When the evaluations are complete, follow these steps:

1. **Compile the individual lists of problems into a single list, and then decide on how to arrive at the severity ratings**. There are several techniques for coming up with severity ratings. If your data collection form asked your evaluators to rate the perceived severity of problems, you could take an average of the scores for problems found by multiple evaluators. A better solution is to set up a meeting (face-to-face or remote) to review each problem and ask each evaluator to indicate how severe each problem is. If there is too much variability in severity scores for the same problem (e.g., you have three evaluators—one rates the problem as severe (5), another rates the problem as moderate (3), and yet another rates it as minor (1)), you can discuss the differences and arrive at a consensus rating. This type of discussion will lead to a better understanding of issues within your product purview and sharpen the skills of the evaluators.
2. **Organize the problems in a way that is useful to the product team**. For example, you can highlight the most serious problems in an executive summary and then organize individual problems by their location in the product. A useful activity before you conduct a heuristic evaluation is to show a sample report to the product team (especially the development manager) and get their feedback on the usefulness and usability of the report. Heuristic evaluation reports sometimes recommend potential solutions for the problems that emerged. Whether you make general or concrete recommendations sometimes depends on your role and the politics of your organization. If you have a design team, you may be pressured to explain the problem, but not suggest anything more than a very general solution because that is the design team's responsibility. They are the designers—you are the evaluator.

3. **Consider whether you want to have a group meeting of the evaluators, developers, and designers to prioritize the results and discuss recommended solutions to the problems**.
4. **Catalog, prioritize, and assign problems, themes, and issues to the appropriate members of the product team**. Arrange subsequent meetings to review potential solutions for the important problems.
5. **Validate the changes to the product with user tests, beta feedback, or other evaluation methods whenever possible**.

VARIATIONS AND EXTENSIONS OF THE HEURISTIC EVALUATION METHOD

This section describes variations and extensions of the heuristic evaluation method from the research and practitioner literature.

The Group Heuristic Evaluation with Minimal Preparation

With the widespread move to agile development, some UX and agile teams are doing group heuristic evaluations where a team of five to ten UX, domain, and product experts review a feature or set of features against a set of heuristics during a single group session. Generally, the product owner or the designer of a feature will walkthrough through common tasks and the evaluators will call out violations of heuristics. The problems are noted by recorder. Questions for clarification are allowed, but not discussion about design solutions. These group heuristic evaluations can last one to three hours (three hours is about the limit of concentration). Often, a second session with a core group is used to discuss factors (e.g., frequency, extent, impact) affecting the severity of the potential problems.

Crowdsourced Heuristic Evaluation

With current collaboration tools, it is possible to expand the scope of a heuristic evaluation to include dozens of participants. You can provide users and UX practitioners with prototypes online and ask them to describe problems, the severity of each problem, and the rationale (e.g., the heuristic) for each problem. At the time of this writing, there seems to be relatively little academic discussion of crowdsourced heuristic evaluation.

Participatory Heuristic Evaluation

Muller, Matheson, Page, and Gallup (1998, p. 14) extend the basic heuristic evaluation method by adding a new category of heuristics called "Task and Work Support." In addition, Muller included users (called "work-domain experts") in the evaluation and suggested that participatory heuristic evaluation (PHE) can be nearly as cost-effective as heuristic evaluation if users are close by and easy to recruit. Muller and his colleagues discuss how the earlier sets of heuristics were generally product oriented and concerned with problems in isolation. The task and work support heuristics focus on user goals (produce quality work, keep sensitive material private, enhance my skills) and a positive experience in the workplace.

Task and Work Support Heuristics (Muller et al., 1998)

Skills. The system supports, extends, supplements, or enhances the user's skills, background knowledge, and expertise. The system does not replace them. Wizards support, extend, or execute decisions made by users.

Pleasurable and respectful interaction with the user. The user's interactions with the system enhance the quality of her or his experience. The user is treated with respect. The design reflects the user's professional role, personal identity, or intention. The design is aesthetically pleasing—with an appropriate balance of artistic as well as functional value.

Quality work. The system supports the user in delivering quality work to her or his clients (if appropriate). Attributes of quality work include timeliness, accuracy, aesthetic appeal, and appropriate levels of completeness.

Privacy. The system helps the user to protect personal or private information—belonging to the user or to her or his clients.

Cooperative Evaluation

Monk, Wright, Haber, and Davenport (1993) published a procedural guide to a technique they called "cooperative evaluation." Cooperative evaluation involves pairing a user and designer in an evaluation of a working version of a product. In the cooperative evaluation, users can freely ask questions of the designer, and the designer can ask questions of the user. The method, like early versions of heuristic evaluation, is aimed at designers with limited human factors or UX backgrounds.

Heuristic Walkthrough

Sears (1997) developed a technique called a "heuristic walkthrough" that had some of the attributes of three UCD methods: (1) a heuristic evaluation, (2) a perspective-based inspection, and (3) a cognitive walkthrough. In Sears's method, the evaluators were given a prioritized list of user tasks, a set of heuristics, and "thought-provoking questions" derived from the cognitive walkthrough method described in Chapter 3.

Sears divided the walkthrough into two phases. Phase 1 required evaluators to conduct a task-based review that involved working through the tasks and the thought-provoking questions for the tasks. Phase 2 involved free exploration of the UI and an evaluation against a list of usability heuristics. Sears found that this hybrid method yielded fewer false positives than a heuristic evaluation and more real problems than the cognitive walkthrough method. The primary difference between the heuristic walkthrough and the two-pass heuristic evaluation (where the inspectors use task scenarios in one pass and then do either guided or free exploration of the UI) is the inclusion of the thought-provoking questions.

HE-Plus Method

Chattratichart and Lindgaard (2008) describe a variation on the heuristic evaluation method called HE-Plus. HE-Plus uses the heuristics described by Nielsen (1993), but also adds a "usability problems profile" that lists common problem areas for the system being evaluated. Here is an example of a usability problem profile from the 2008 paper that shows eight problem areas for a hotel reservation site (Chattratichart & Lindgaard, 2008, p. 2216).

1. Information content
2. Navigation
3. Graphics and UI objects
4. System features, functionality, defaults, interaction with I/O devices
5. Formatting and layout
6. Wording and language
7. Feedback, help, and error messages
8. Pan-disability accessibility.

The authors provide some research support for the superiority of the HE-Plus method that combines both heuristics and a usability problem profile. The practitioner takeaway here is that the combination of heuristics and a list of common problem areas for the type of software you are focused on can improve the quality of your heuristic evaluation.

MAJOR ISSUES IN THE USE OF THE HEURISTIC EVALUATION METHOD

The heuristic evaluation method has seen much scrutiny since the early 1990s. Some of the major issues that will face practitioners are described in the section below.

How Does the UX Team Generate Heuristics When the Basic Set Is Not Sufficient?

While a detailed discussion about how to generate custom heuristics is beyond the scope of this book, a few brief notes might be helpful. There are several ways to generate heuristics when the basic sets are not sufficient. First and most efficient, you can search the literature for heuristics that have been used to evaluate the usability of products and services similar to yours. The ACM Digital Library (http://dl.acm.org) is an excellent source of heuristics dealing with a wide range of products and particular areas such as accessibility, security, game design, ambient displays, information architecture, and persuasive technologies. The ACM research literature often describes how new heuristics are generated and validated (usually by examining how well the new heuristics support the identification of known problems in related products and services).

A second approach (Mankoff, Dey, Hsieh, Kientz, Lederer, & Ames, 2003) is to develop a list of the primary goals of a system, see how well standard lists of heuristics meet the primary goals of a system, remove irrelevant heuristics, and modify relevant ones to focus on the goals of your target system. Brainstorm new heuristics with experts, and then have an additional group of designers and domain experts review and refine the heuristics using a workshop or survey method. After developing a new set of heuristics, you can test them against a known system to validate the efficacy.

Do Heuristic Evaluations Find "Real" Problems?

There is a major debate in the field about what constitutes a usability problem (Wilson, 2007). Bailey, Allan, and Raiello (1992) suggested that many of the "problems" found in heuristic evaluations were not actually performance or preference problems at all (they called these "false positives"). Other researchers have noted that heuristic evaluations miss many usability problems. Doubleday, Ryan, Springett, and Sutcliffe (1997) compared the problems found in a heuristic evaluation using five UCD experts with the problems that emerged from end-user testing.

Doubleday and her colleagues found that the heuristic evaluation found only fifteen of the thirty-eight problems (39%) that occurred during end-user testing. In comparison, the end users experienced only 40% of the problems predicted in the heuristic evaluation. Following are the possible reasons for these differences:

- Heuristic evaluators, even when working with task scenarios, are not immersed in the task in the same way as end users.
- Heuristic evaluators who are experts in areas such as Windows or web interaction styles may not experience some very basic problems that can lead to confused users. For example, elderly users, unlike most UCD experts, may have no idea that they should avoid clicking on advertising buttons that say "Click here to get rid of viruses." Elderly users who are new to computing and the web may think that they *must* click on those buttons. UCD experts might not be able to easily emulate the issues and problems experienced by particular populations (brand-new users, kids, multitasking users who are trying to talk on the phone, elderly users, physically/mentally challenged users).
- The granularity of the problems in the heuristic evaluations differed. Some evaluators specified very general problems such as "missing shortcuts," whereas others listed "there is no shortcut for the Browse feature."

The definition of "problem" used in these studies is often "something that was found in a usability test with actual users." Studies that examine the validity and reliability of heuristic evaluations often base the efficacy of heuristic evaluation on a comparison with the problem list derived from usability testing. However, user testing is not a foolproof method for generating the "true and complete" problem lists for anything beyond extremely simple UI objects. The results of usability testing are affected by facilitation experience, the particular tasks chosen, the choice of and background of participants, the size of the database, and other factors. Cockton et al. (2012) discuss many issues associated with the reliability and validity of usability inspection methods, including heuristic evaluations.

Triangulation of Problems

The most powerful method for determining what the "real" problems are is triangulation (Wilson, 2006), in which you use multiple methods (e.g., cognitive walkthroughs, heuristic evaluations, user testing, feedback from customers on prior versions of the product, reviews of technical support call records, and participatory design sessions) to find the problems with a product. Problems that are found repeatedly using different methods with different users on different systems are the real problems that should receive the highest priority in the development schedule.

Does Heuristic Evaluation Lead to Better Products?

Heuristic evaluations might be the most-used usability method (Rosenbaum et al., 2000), but the connection between finding problems and measurable improvements in products is difficult to establish. To establish this connection, you need to do the following:

- Find problems.
- Come up with a good solution for each problem.
- Persuade developers to implement the solution as you recommended.
- Test the final product taking into account that some fixes will introduce completely new problems.

How Much Does Expertise Matter?

One of the consistent findings of the research on various forms of usability evaluation methods is that expertise does matter. Three major types of expertise affect the results of heuristic evaluations: knowledge of the heuristic evaluation method and principles, knowledge of the domain, and knowledge of the expected context of use (Cockton et al., 2002, p. 1127). Lack of knowledge in these three areas can lead to increased false positives and missed problems. The following are methods of improving expertise:

- Conduct a training session with a focus on the set of heuristics that you are using and definitions of problems and severity levels.
- Sponsor some seminars on design topics to prime potential evaluators if you are working in a corporate environment.
- Use domain-specific heuristics, which can be developed from usability testing or competitive analysis and can be used to improve the thoroughness of heuristic evaluations. The drawback with domain-specific heuristics is that it takes time and domain expertise to develop and validate them.

- Provide some context-of-use information for the team. For example, you might discuss common scenarios, user profiles, and background information on the main issues for this type of product.
- Invite domain experts to be part of your review team and, if possible, have double experts who have both usability and domain knowledge.

Should Inspections and Walkthroughs Highlight Positive Aspects of a Product's UI?

Inspections and walkthroughs are heavily focused on problems and seldom highlight positive aspects of a product's UI. A guideline for usability test reports is that they highlight positive aspects of the product as well as negative aspects; heuristic evaluation reports can also highlight the major positive aspects of a product. Listing the positive aspects of a product has several advantages:

- Evaluation reports that highlight positive and negative issues are perceived as more balanced by the product team.
- You might reduce the likelihood that something that works well is changed for the worse.
- You may want to propagate some of the positive design features throughout the product.
- Sometimes the positive features being mentioned actually bring focus to some of the very negative features being highlighted.

Individual Reliability and Group Thoroughness

The unreliability of individual problem reports, where different individuals report different problems, is well documented and is the rationale for having multiple inspectors. Hartson, Andre, and Williges (2003) point out that attempts to standardize inspection processes (by including detailed tasks and procedures, for example) might improve the reliability of multiple inspectors but reduce the thoroughness of the group results by "pointing all the inspectors down the same path" (p. 168).

DATA, ANALYSIS, AND REPORTING

The primary data from a heuristic evaluation are a list of problems, each categorized as follows:

- Location in the UI
- A description of the problem

- Which heuristics are associated with the problem
- The number of evaluators who found the problem. The type of evaluator (e.g., usability practitioner versus developer versus domain expert)
- Whether the problem is a global issue (it appears in many places in the product) or a local problem that is confined to a single location (Dumas & Redish, 1999)
- Severity rating from each evaluator to assist in prioritization of problems (rating can be the result of discussion and consensus or an average of the individual ratings).

Given that heuristic evaluations are meant to be efficient, the immediate analysis is generally a list of problems that are prioritized by criteria important to the product team (e.g., severity, cost, difficult to implement). The "keeper" of the data from heuristic evaluations and other sources of usability problem data can track data over time and examine the data for patterns:

- Are particular heuristic being violated often?
- Are severity ratings becoming more consistent over time?
- Who are the best evaluators?
- Do evaluators find different problems as a result of their role or experience?
- Are the particular areas of a product where more problems than expected are being found?
- Do you need to consider additional heuristics?

Yahuda and McGinn (2007) presented the results of a heuristic evaluation of eight websites by defining "usability expectations" for features of interest (e.g., contact information). The intent of this research was to determine which solutions for features across the eight sites were the best based on how well the solutions met usability expectations. These usability expectations were based on a set of heuristics and were focused on the domain of interest. Each usability expectation was assigned a value based on a maximum of 5 points. Evaluators rated a set of designated features by assigning points to the usability expectations. This method can be used for example, to rate a "Directions" feature. The Directions feature might receive 1 point for being usable on a small screen, 1 point for having text that is readable, 1.5 points for clear verbal driving instructions from your location, 1 point for simplicity (not too much detail), and 0.5 points for providing useful landmarks—for a total of 5 points. The sum of the points for each feature in the Yehuda and McGinn study was reported on a five-star rating. Features that met all usability expectations were rated as ★ ★ ★ ★ ★. Features that met around 60% of the usability expectations received a ★ ★ ★ rating. Features that failed to meet most or all of the expectations got a ★ rating (an "awful") solution. The star ratings as well as the ratings for each usability expectation provided quick assessments of feature usability as well as the details behind the star ratings.

One of the questions that you might want to consider is "How do you measure the success of a heuristic evaluation?" In the usability literature, the focus is generally on how many problems of various severities are found. That is a start, but a more important measure might be how many problems are fixed. Sawyer, Flanders, and Wixon (1996) suggest that the results of heuristic evaluations and other types of inspections look at the impact ratio which is the ratio of the number of problems that the product team commits to fix divided by the total numbers of problems found times 100 (p. 377). While the impact ratio provides a measure of how many problems are fixed, this measure still does not indicate how much more usable the product is as a result of the identified and fixed problems.

CONCLUSIONS

The heuristic evaluation method has received much attention since it first emerged in the early 1990s as a discount usability technique. The technique can be used by an individual or a multidisciplinary team. This chapter highlighted both the strengths and weaknesses of the heuristic evaluation method and also provided numerous tips for improving the quality of heuristic reviews. With appropriate training, examples, and knowledge of users, tasks, and environments, heuristic evaluation will be a useful tool for user experience practitioners.