Chapter 3

Cognitive walkthrough and heuristic evaluation

3.1 Overview

In the previous chapter, the preliminary design of the web based CO2SYS program was discussed. This chapter will talk about how to evaluate this preliminary design with two simplified methods: cognitive walkthrough and heuristic evaluation.

3.2 Informal system testing without real users

Usability of a web page interface is very important for users. They can get frustrated, keep making mistakes, waste their time and be unable to finish tasks. So usability evaluations are critical to the development processes of the interface. The field of human-computer interaction (HCI) recommends conducting usability evaluations early in the design process, especially in the stages of early prototyping (Rieman et al., 1995). If the critical flaws can be found at an early stage, they also have higher chances to be found and fixed due to the relative simplicity of the system. When the project gets more and more complicated at later stage, it will be extremely difficult to find and fix the problems. So usability evaluations at an early stage save lot of time and effort for the project in the long run. Empirical user testing is the most comprehensive evaluation technique, which is expensive and requires a lot of time and the involvement of real users. To save time and cost, most developers only use this technique at the later stage or the end of the design cycle. Sometimes the changes are very costly and difficult to make, so the recommendation from users are often ignored unfortunately.
Here we introduce two types of methods that can be used to evaluate an interface without real users. The first one is “Cognitive walkthrough”, a task oriented method for evaluating the design of a user interface and checking how well the interface can support first time use without formal training (i.e., exploratory learning) (Polson et al., 1992; Rieman et al., 1995; Wharton, et al., 1992). This evaluation can be performed by the system designer at the early stage of design. The second one is “Heuristic evaluation”, a technique used to check how well the interface complies with a list of general principles and guidelines for interface design.

### 3.3 Cognitive walkthrough

Cognitive walkthrough has two phases: preparation and evaluation. For the preparation phase, four kinds of information have to be ready before the walkthrough starts: 1.) A preliminary description or prototype of this interface. It does not have to be complete, but it should include as many details as possible. As we described in Section 2.4, a preliminary design of the web page layout is ready for the evaluation. 2.) A list of tasks that should represent those used for the task-centered design of the system. Based on the typical four scenarios in Section 2.3, four main tasks are proposed here for the walkthrough evaluation:

i.) Given two parameters (TA and TC), calculate the other two parameters (pH and $f_{CO_2}$) for freshwater under single mode

ii.) Given two parameters (TA and TC), calculate the other two parameters (pH and $f_{CO_2}$) for freshwater under batch mode
iii.) Given two parameters (TA and TC), calculate the other two parameters (pH and $f_{CO_2}$) for seawater under single mode

iv.) Given two parameters (TA and TC), calculate the other two parameters (pH and $f_{CO_2}$) for seawater under batch mode

3.) A complete, written list of the actions used to complete each individual task on the interface. 4.) An idea of who the users will be and what kind of experience they will have when they first use the system, which is discussed in Section 2.2. This understanding should be developed through the task and user analysis.

After the preparation phase is complete, the designer can start the cognitive walkthrough process. During the walkthrough process, the designer tries to tell a story about the prospective user’s motivation and examines each individual step in the correct action sequence. To critique if the story is believable or not, four questions below can be asked to help the designer evaluate whether the user will choose the correct action.

1.) Will users be trying to produce whatever effect the action has?

2.) Will users be able to locate the control (button, menu, switch, etc.) for the action?

3.) Will the control users found help them reach the goal they want?

4.) Will the interface provide understandable feedback for users after each action?

The first question above talks about what the user is thinking. Most of the time users do not think or act the way the designer expects them to. If users want to calculate for freshwater sample under the batch mode, will they set the water type to “Freshwater” and the input mode to “Batch mode” before they run the calculations? The second question deals with the difficulty of finding the right control by the user. Sometimes it is
really hard to find the control, not to mention the right control. Will users know the right controls to set the water type and input mode for any specific data calculations, for example? The third question concerns the identification of the control. If users find the control, can they tell this is the right control that can implement the tasks they want? These three questions are interactive. For example, a clearly labeled control may remind users what needs to be done even if they might not do the right thing initially. The final question asks about the feedback after the action is complete. Users need to see the feedback from the system, even the simplest, so they will know the system has taken the action. The detailed walkthrough information including the listed actions for each task is described in Appendix A.

After the walkthrough with the main tasks above on the preliminary prototype, we found that this version of design passed most of the evaluations at this development stage. The only problem we caught during this walkthrough process is that there is no feedback from the system after the user clicks the “Execute” button to run the calculation under the batch mode, as shown in Figure 3.1 and in Appendix A. After the calculation is complete, the system should give a message about the status of the calculation for the user. The solution for this problem will be discussed in the next chapter.

3.4 Heuristic evaluation

Heuristic evaluation, mainly proposed by Nielsen (1989), and Nielsen and Molich (1990), are general guidelines or principles used to guide interface designs. It is inexpensive relative to other evaluation methods and easy to use. The evaluators do not have to have formal training and advanced planning is not required.
Heuristic evaluation is a user-centered method, which is different from the cognitive walkthrough as a task-oriented technique. Task-oriented techniques can detect interface problems occurring while the user tries to finish the tasks and describe the
importance of the problems under the live situation. But it is impossible to evaluate every
task the user will perform. Each user may have a different sequence of actions and
control executions. Moreover, each task is evaluated separately and the cross-task
interactions are not identified. So this task-free user-centered method, heuristic
evaluation, is brought in to catch problems that cognitive walkthrough may have missed.

Based on their own experience, Nielsen and Molich (1990) have summarized nine
heuristics for this evaluation method:

1. Simple and natural dialog: the interface should not have irrelevant or rarely
used information. All the information should be arranged to match the task.
2. Speak the user’s language: system-specific engineering terms should be
avoided and concepts from the user’s world should be considered.
3. Minimize user memory load: keep the information on the screen until not
needed so the user does not have to remember things from one action to
another.
4. Be consistent: one action sequence the user learned from one place should
also apply in another place.
5. Provide feedback: give the user clear message of the effect from their actions.
6. Provide clearly marked exits: the user should be able to enter or exit from any
part of the system without any difficulties or damages to the system.
7. Provide shortcuts: shortcuts can help experienced users with efficiency by
bypassing unnecessary steps and information.
8. Good error messages: they can tell the user what the problem is and how to
correct it.
9. Prevent errors: help the user avoid making mistakes and save their time and effort.

The four main tasks used in Section 3.3 for the cognitive walkthrough are evaluated with these nine heuristics again. The detailed results for this evaluation are included in Appendix B. Most of the preliminary design fits the user interface standards. There are two main problems found from this evaluation. There is no feedback from the interface when data calculations are performed under batch mode, which is the same as the one found from the cognitive walkthrough process. Secondly, lack of error checking for the user input makes the program unable to handle the errors for the user as shown in Figure 3.2. When the user enters an invalid value, the system should generate an error message to help the user fix the error. Useful information for each input should be provided to help the user from entering invalid inputs. The detailed solutions to these two problems will be addressed at next stage of the development.

3.5 Summary

Two types of evaluation methods, cognitive walkthrough and heuristic evaluation, were used by the designer to evaluate the preliminary design of the web-based CO2SYS program. Since the system is still at a premature stage and it is not totally functioning yet. The problems found from these two evaluation processes would be fixed during further implementation from the preliminary design of the system. Real users would be brought in to test the mature version of the program, which is discussed in the next chapter.
Figure 3.2 The web page layout of data calculations for freshwater sample under the single mode after the user enters an invalid value for the input temperature. The invalid input is marked with a red circle.