silly musical effects when erroneous choices or erroneous inputs were made.

One has to wonder why Silicon Techtronics did not attempt a more sophisticated approach to the interface design. After a careful study of the Robbie CX30 applications domain, I have come to the conclusion that a direct manipulation interface, which literally displayed the robot at the operator console, would have been ideal. The very visual domain that the robot operated within would lend itself naturally to the design of appropriate screen metaphors for that environment, metaphors which the operator could easily understand. This would allow the operator to manipulate the robot by manipulating the graphical representation of the robot in its environment at the computer console. I have asked one of my doctoral students, Susan Farnsworth, to give up her personal life for the better part of a decade in order to investigate this possibility a bit further.

4. How the Robbie CX30 interface violated the eight golden rules

The Robbie CX30 user interface violated each and every golden rule in multitudinous ways. I shall only discuss a few instances of rule violation in this paper, leaving a more detailed discussion of these violations for future articles and my forthcoming book1. I will emphasize those violations which were relevant to this particular accident.

4.1 Strive for consistency

There were many violations of consistency in the Robbie CX30 user interface. Error messages could appear in almost any color and could be accompanied by almost any kind of musical effect. Error messages could appear almost anywhere at the screen.

When Bart Matthews saw the error message for the exceptional condition which occurred, an exceptional condition which required operator intervention, it was probably the first time he saw that particular message. In addition, the error message appeared in a green box, without any audio effects. This is the only error message in the entire system which appears in green and without some kind of orchestral accompaniment.

4.2 Enable frequent users to use shortcuts

This principle does not appear in any way in the entire interface design. For example, it would have been a good idea to allow frequent users to enter the first letter of a submenu or menu choice in lieu of requiring the use of the cursor keys and the enter key to effect a menu choice. The menu selection mechanism in this system must have been quite a mental strain on the operator.

Furthermore, a form of type-ahead should have been supported, which would have allowed a frequent user to enter a sequence of menu choices without having to wait for the actual menus to appear.

4.3 Offer informative feedback

In many cases, the user has no idea whether a command that was entered is being processed. This problem is exaggerated by inconsistencies in the user interface design. In some cases the operator is given detailed feedback concerning what the robot is doing. In other cases the system is mysteriously silent. In general, the user is led to expect feedback and consequently becomes confused when no feedback is given. There is no visual representation of the robot and its environment at the screen and the operator's view of the robot is sometimes obstructed.

4.4 Design dialogues to yield closure

There are many cases in which a given sequence of keystrokes represents one holistic idea, one complete task, but the operator is left without the kind of feedback which would confirm that the task has been completed. For example, there is a fairly complicated dialogue which is necessary in order to remove a widget from the acid bath. However, upon completion of this dialogue, the user is led into a new, unrelated dialogue, without being informed that the widget removal dialogue has been completed.

4.5 Offer simple error handling

The system seems to be designed to make the user regret any erroneous input. Not only does the system allow numerous opportunities for error, but when an error actually occurs, it is something that is not likely to be repeated for some time. This is because the user interface makes recovery from an error a tedious, frustrating and at times infuriating ordeal. Some of the error messages were downright offensive and condescending.

4.6 Permit easy reversal of actions

As mentioned in the previous paragraph, the user interface makes it very difficult to recover from erroneous inputs. In general, the menu system does allow easy reversal of actions, but this philosophy is not carried through to the design of dialogue boxes and to the handling of exceptional conditions. From a practical (as opposed to theoretical) point of view, most actions are irreversible when the system is in an exceptional state, and this helped lead to the killer robot tragedy.

4.7 Support internal locus of control

Many of the deficiencies discussed in the previous paragraphs diminished the feeling of "internal locus of control". For example, not receiving feedback, not bringing interactions to closure, not allowing easy reversal of actions when exceptions arose, all of these things act to diminish the user's feeling of being in control of the robot. There were many features of this interface which make the operator feel that there is an enormous gap between the operator console and the robot itself, whereas a good interface design would have made the user interface transparent and would have given the robot operator a feeling of being in direct contact with the robot. In one case, I commanded the robot to move a widget from the acid bath to the drying chamber and it took 20 seconds before the robot seemed to respond. Thus, I did not feel like I was controlling the robot. The robot's delayed response along with the lack of informative feedback at the computer screen made me feel that the robot was an autonomous agent - an unsettling feeling to say the least.

4.8 Reduce short-term memory load

A menu driven system is generally good in terms of the memory burden it places upon users. However, there is a great variation among particular implementations of menu systems insofar as memory burden is concerned. The Robbie CX30 user interface had very large menus without any obvious internal organization. These place a great burden upon the operator in terms of memory and also in terms of scan time, the time it takes the operator to locate a particular menu choice.

Many dialogue boxes required the user to enter part numbers, file names, and other information from the keyboard.