EMPIRICAL STUDIES
RECHERCHES EMPIRIQUES

TRAINING OF SYSTEMATIC DIAGNOSIS:
A CASE STUDY IN ELECTRONICS TROUBLESHOOTING

by J. M. C. Schraagen* and A. M. Schaafstal*

I. INTRODUCTION

The study reported in this paper was aimed at developing better methods for training naval troubleshooters. To accomplish this goal, two experiments were carried out. The first experiment, based on theoretical...
notions about diagnostic skill investigated the necessary knowledge and skills needed to carry out diagnosis in an electronics domain, as well as the problems novices encounter in troubleshooting. As a result of this exploratory study, hypotheses were formed about the type of knowledge and strategies that are lacking in novice maintainers, and how they could possibly be trained. The second experiment consisted of a more formal evaluation of these ideas, implemented in a knowledge-based environment for troubleshooting. The knowledge-based environment was used as a tool to test particular hypotheses about the effectiveness of various types of support knowledge. If a particular type of knowledge or strategy is found to be effective in enhancing troubleshooting performance, then this knowledge or this strategy may be taught to novice maintainers in various ways, for instance by traditional classroom instruction, by practising faultfinding or by intelligent tutoring systems. After a brief outline of the theoretical notion about diagnostic skill on which the study is based, the two experiments will be discussed.

II. TRAINING OF DIAGNOSTIC SKILL:
THEORETICAL NOTIONS

Many researchers make a distinction between declarative and procedural knowledge (e.g., Anderson, 1983, 1987). Declarative knowledge may be conceived of as a collection of stored facts and is also called system or device knowledge in the domain of technical systems. Examples are knowledge about normal values of certain parameters, or knowledge about the function of the system. Procedural knowledge (knowledge about how-to-do-it) can be regarded as a collection of actions or procedures that an intelligent system can carry out. It also consists of knowledge of the procedures with which one investigates a device to make diagnoses about its dysfunctioning, for example the use of the oscilloscope to test certain functions of a system. Procedural knowledge is content- and context-specific and thus only applicable in a limited domain.

In addition to the declarative-procedural distinction, a distinction can be made between domain-specific knowledge and strategic or metacognitive knowledge. This strategic or metacognitive knowledge (knowledge about how-to-decide-what-to-do-and-when) is applicable across specific content domains, but remains geared towards one task (e.g., diagnosis). For example, in diagnosis, regardless of the domain, one would first identify and interpret symptoms, followed by an investigation of possible reasons, which will be tested, before one will apply a certain repair or remedy. This level of strategic knowledge, task specific but domain independent, will be referred to in the remaining of this paper as the task structure of a diagnostic task. Empirical evidence for such a task structure was obtained by Schaafstal (1993) in her studies of expert and novice operators in a paper mill.

The task structure of a diagnostic task should be contrasted with problem-solving strategies which are concerned with very general thinking and reasoning skills, such as means-ends analysis, reasoning by analogy, or
Training of systematic diagnosis

working backwards. These general problem-solving strategies are applicable across specific tasks, and are thus of a more general character than the task structure, which is task specific.

With this decomposition, it is assumed that procedural and declarative knowledge are organized and deployed by goals, plans and decision rules that comprise strategic knowledge. Thus, strategic knowledge can be said to have a control function to enable dynamic, flexible reasoning. As described in Gott (1989), support for the concept of strategic control knowledge comes from a number of domains such as text editing (Card, Moran, & Newell, 1983), computer programming (Anderson, Boyle, Farrell, & Reiser, 1987), and simple device operation (Kieras & Bovair, 1984).

One of the most striking findings in the literature with respect to the training of cognitive skill, as summarized in Morris and Rouse (1985), is that instruction in theoretical principles is not an effective way to produce good troubleshooters.

However, when theoretical instruction is combined with training people in how to use that knowledge, performance usually gets better. For example, Miller (1975), as described in Morris and Rouse (1985), developed an experimental training course for radar mechanics, in which an effort was made to relate instruction in system functioning to actions performed during trouble-shooting. For example, system behavior was presented in terms of causal sequences rather than a more traditional presentation of schematics. Whenever possible, attempts were made to relate schematics to the actual equipment. A control group received instruction in the theory upon which the system was based and a left-to-right presentation of schematics, with no special attempt to relate the information to the actual equipment. Both groups had limited access to the actual radar for troubleshooting practice. The experimental group turned out to be faster in performing checks and adjustments, was more often successful in troubleshooting, made fewer errors in general, and scored better on quizzes.

It should be noted, though, that in the studies in which positive effects were found, the guidance involved was rather explicit: students were told to generate hypotheses, chunk information, and analyze symptoms in a prescribed way, which comes very close to training to use a task structure. This is a far more active approach than just providing an opportunity to use system knowledge, and should not be interpreted as evidence that the latter approach will produce better troubleshooters.

The review by Morris and Rouse (1985) should, however, not be interpreted to mean that all theory is useless. As Patrick (1993) pointed out, some theory may be better than other theory. For instance, interpretation of structural diagrams, utilisation of principles of data flow, and causal models concerning plant variables and components (Patrick & Haines, 1988) may actually be helpful to fault-finding. What is presumably less useful is a too detailed explanation of how a system works. By "too detailed" we mean "more detailed than is actually required for fault-finding". In practice, troubleshooting often stops when a faulty component is detected and replaced. It is of little or no use for troubleshooters to understand the system beyond the level of those faulty components or "Line Replaceable Units" (LRU's) as they are sometimes called. Therefore, a functional
understanding of the system is often required for troubleshooting but a detailed understanding of how each function is realized physically is often not required.

Another important factor in training of troubleshooting is the opportunity for practice (e.g., Johnson & Rouse, 1982a, 1982b), which is also strongly advocated by many researchers in intelligent tutoring systems (e.g., Lesgold, 1992).

Training with respect to task structures is still a rather neglected area. There may be a number of reasons for this neglect. First, to be able to train strategies, these strategies should first be made explicit, which is not an easy task to accomplish. Second, the training of strategies requires a learning environment with sufficient possibilities for practice, and guidance in using a good strategy. Only now, these environments become available, often as a result of the application of AI-oriented research. Third, the relationship between the training of domain knowledge (e.g., "theory") and the training of strategies how to apply that knowledge is not yet very well understood. Therefore, it is very logical that in many intelligent tutoring systems an emphasis is put on either domain knowledge (e.g., STEAMER: Hollan, Hutchins, & Weitzman, 1987), strategic knowledge (e.g., LISP TUTOR: Anderson & Reiser, 1985), or even completely other aspects such as dialogue issues.

One of the problems that occurs with respect to the evaluation of ideas brought into an intelligent tutoring system is that often there are so many of them that, although the system in general may show very promising results, it is unclear where the effect comes from: is it the adequacy of implementation of system knowledge, is it the strategy that improves performance or are there other factors that contribute to the success of a particular system? To enable a more systematic evaluation of the success of individual factors, one would need several versions of a system, which may not always be feasible. Therefore, as will be discussed later on in this paper, for the second experiment we actually built two versions of a system, one supporting only domain knowledge, the second one supporting both domain knowledge and a troubleshooting strategy. These two situations are compared with a control situation in which people have to troubleshoot problems with the system documentation as only help available.

III. EXPERIMENT 1: ELECTRONICS TROUBLESHOOTING IN A RADAR SYSTEM

The first experiment reported in this paper was aimed at characterizing the differences between novices and experts in the field of troubleshooting in a radar system. Therefore, six subjects, with varying levels of knowledge and expertise in this particular radar system, were asked to solve problems in an actual radar system, which was located in a training facility of the Royal Netherlands Navy. Four subjects worked alone, two subjects worked as a team (the data of the team will be
Training of systematic diagnosis

considered together). Each subject had to diagnose the same two malfunctions, which were selected by school personnel and rated as being of moderate difficulty. The first malfunction was a fault in the high voltage stabilization circuit of the sender part of the radar, caused by a faulty fuse. The second malfunction was a fault in the receiver caused by a faulty cable. The subjects were allowed to use all available documentation, and could ask for specific help in case they were stuck somewhere in the problem. The help was provided by the school personnel. Subjects were asked to think aloud during the experiment. Written notes were made in order to collect the protocols. All documentation used and help given was also written down on paper.

The data analysis started with an investigation of the task structure used by the subjects, and the ease with which parts of the problem solving process, reflected by the thinking aloud protocol, could be assigned to elements of the task structure. Since it was impossible for the subjects to apply repairs, the following task structure was defined beforehand:

- identify symptom (e.g., observe status indication);
- interpret symptom (e.g., Helix current too large);
- generate possible causes (e.g., failure in Helix test circuit);
- test possible causes (e.g., measure Helix current).

Each element in the task structure was operationalized by recording the local strategies subjects used to carry out the global steps in the task structure. The particular local strategies used are dependent on the domain knowledge available. When sufficient domain knowledge is available, symptomatic search (a recognition-based strategy, see Rasmussen, 1984) will be used when generating possible causes. When domain knowledge is partly available, functional or topographic search (Rasmussen, 1984) will have to be used. This search strategy employs a representation of a system’s functions or their location. Finally, when domain knowledge is insufficient, system documentation or fault finding procedures will have to be consulted. Fault finding procedures consist of binary-choice fault finding trees provided by the system manufacturer. For each step in the task structure, we inferred from the verbal protocols whether subjects used functional and topographic search (e.g., “the signal at the end of the Travelling Wave Tube is correct, therefore the high-voltage stabilisation circuit must be faulty”), whether they measured something (e.g., “let’s measure on board A27, position 94”), and whether they consulted system documentation and fault finding procedures. The number of times these strategies were used was counted and was expressed as a proportion of the total number of steps taken. Finally, this proportion was related to success in fault finding, expressed as a dichotomous measure (successful-not successful).

The results showed that almost all of the protocol segments could be assigned to the elements of the task structure with very high agreement about the elements (>90%) between two independent raters. Malfunction number 1 was only solved correctly by the team. Malfunction number 2 was solved by the team and by two of the four individuals. Hence, there was a success rate of 40%. Table I shows, averaged across subjects and faults,
the average number of times per step in the task structure that a particular
local strategy was used, for both the malfunctions solved and not solved.
Due to the small number of subjects, significance tests were not performed,
and results should merely be taken as indicative, to be tested more
thoroughly in Experiment 2.

Table I shows that the successful troubleshooters less frequently
consulted system documentation and fault finding procedures than the
unsuccessful troubleshooters. On the other hand, they made more use of the
strategy of functional and topographic search, and they made more
measurements (the larger number of measurements was almost exclusively
due to the team consisting of two troubleshooters working together; this is
not surprising because the team worked more efficiently than the
individuals). Although these results are of a correlational nature, they
confirm our intuition that knowledge of the structure and function of a
system may contribute to success in troubleshooting. In the absence of such
knowledge, troubleshooters have to resort to the available documentation
which results in slower and more laborious troubleshooting. Because there
was a time limit set at 1.5 hour for each malfunction, it is not surprising that
when troubleshooters have to rely on system documentation, they will not
solve the problem within the time available.

Table I
Average number of times per step in the task structure that a local strategy is used
Nombre moyen d'occurrences d'une strategie locale par etape de la procedure

<table>
<thead>
<tr>
<th>Strategies used</th>
<th>Solved</th>
<th>Not solved</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td>.04</td>
<td>.19</td>
</tr>
<tr>
<td>Fault finding procedures</td>
<td>.09</td>
<td>.15</td>
</tr>
<tr>
<td>Functional search</td>
<td>.27</td>
<td>.08</td>
</tr>
<tr>
<td>Measurements</td>
<td>.27</td>
<td>.09</td>
</tr>
</tbody>
</table>

The contents of the protocol showed that novice subjects had great
difficulty in generating possible causes, in defining causes at the right level
of abstraction, in knowing how to test certain causes, and in finding answers
in the documentation. They gave the impression of being so busy in solving
all sorts of underlying subproblems that they got lost in the problem, and
therefore were not systematic in their approach anymore.

With respect to the domain knowledge used by the various subjects, the
results showed a wide variety of knowledge used at various levels of
abstraction. The following categories of types of knowledge could be
distinguished (for each category, an illustrative protocol fragment
illustrating use of the knowledge is added):

1 / knowledge of the function of various (sub)systems, e.g., “the function of
the T/R cell is to protect the LNTA”;
2 / knowledge about the location of components, e.g., “B01, where is it?”;
Training of systematic diagnosis

3 / knowledge about normal values or points of reference, e.g., “I expect a continuous signal”;
4 / knowledge about frequency of occurrence of certain causes, e.g., “The TWT almost never breaks down”;
5 / underlying knowledge about electronics, e.g., “If 5kV gets too small then the jet current will not be attracted and the current will be caught by the Helix”;
6 / knowledge about the indexing of the documentation, e.g., “In system documentation part 2, under ‘Operation of the Helix’, one can find what it means if the light I-Helix burns”;
7 / interpreting schematic diagrams, e.g., “what do these symbols mean?”;
8 / knowledge of test procedures, e.g., “how do you test high voltage current?”.

These results are very similar to results obtained by Kurland and Tenney (1988) on troubleshooting performance from maintenance technicians on the AN/MPQ-57 HIPIR Radar. On the basis of these results, the following hypotheses were formed:

— An explicit guidance at the level of the task structure, which will be filled in by the student during a problem solving episode, will help. First, it will keep students on track while solving the problem, and second, it will bring students back on track once they get lost due to working memory overload.

— Subjects will benefit from a rich set of relevant and accessible domain knowledge of various kinds, as described above, and specific to the problem solving phase they are in. The rapid availability of relevant domain knowledge should reduce the troubleshooting time. For example, when they are in the phase of testing causes, subjects should, if needed, be helped with very specific information about testing, such as measurement procedures, values to expect, where to measure and so on. When in the process of generating causes, other types of information may be useful, such as functional schemas.

In addition to these two hypotheses that we wanted to test in a subsequent experiment, we felt that, based on the enormous amount of problems at the level of domain knowledge, just a guidance at the task structure level would not be sufficiently helpful. Therefore, the task structure guidance is always accompanied by help at the domain level. The next experiment was set up to test these hypotheses.

IV. EXPERIMENT 2: TROUBLESHOOTING WITH THE HELP OF A KNOWLEDGE-BASED GUIDANCE ENVIRONMENT

In order to separate out effects of guidance of the task level and guidance of domain knowledge, two versions of a knowledge-based guidance system were developed. The first version provides guidance at both the task structure as well as the level of domain knowledge. The
second version consists solely of guidance at the level of domain knowledge. Both systems are constructed to be rather passive: the student controls the session, is free to do what he wants, and asks for the information he wants to have. However, there is one exception to this: the first version of the system prompts the user to fill in the task structure, and does not allow a violation of the prescribed order of the task structure. For example: a subject cannot test a cause before having generated one, and generating causes is impossible unless a problem description has been made. The idea behind the systems is that they are constructed in such a way so as to be able to fill in the gaps a student might have in his knowledge, and that they help the student to keep track of his problem solving process.

The system is directed towards troubleshooting in a computer system which is in use on board of frigates. Although we would have preferred continuing with the radar system, this turned out to be virtually impossible, due to the very limited number of students who take a course in this system each year and therefore due to the very limited number of available subjects. Therefore, a different system also belonging to the electronics domain was chosen. One of the drawbacks for the project was that quite some time had to be invested by the school to get the researchers sufficiently acquainted with the computer system. However, the pilot experiments carried out with this system showed very similar results as obtained with the radar system, thus giving more confidence in the previously acquired results. In particular, novice troubleshooters in both systems had difficulty maintaining a structured approach and overview of what they were doing. They were also often forced to consult the available documentation due to a lack of functional system knowledge. Hence, there are some general, equipment independent difficulties associated with novice troubleshooting.

IV. 1. Design

The experiment carried out with the guidance systems involved three conditions (between subjects). In condition 1, subjects had to troubleshoot four malfunctions without the help of any system. They had to work on their own, but were allowed to use the available documentation. Condition 2 consisted of troubleshooting the same four malfunctions, the first two without the aid of a system, the final two with the help of the system with only domain knowledge support. The subjects in this condition were only allowed to use the available documentation for the first two faults, the final two were aided just by the system. Condition 3 consisted of troubleshooting the same four malfunctions as in the other conditions, the first two again providing a baseline, the final two aimed at measuring the effect of the system with both domain knowledge and task structure support. Thus, subjects in all three conditions solved the first two faults (fault 1+2) without any system support; differences among the conditions were introduced by the experimental treatment (differential system support) after the first two faults and before the last two faults (fault 3+4). In each condition six subjects participated.
IV. 2. Subjects

The 18 subjects in this experiment just finished a course in the particular computer system, and passed their exam. They had not yet gained practical experience on board of ships in troubleshooting this computer system, but did some troubleshooting exercises during the course. All subjects could be considered novices.

IV. 3. Faults

Subjects were given four unfamiliar faults. In the judgement of experienced instructors, the faults were of average difficulty and representative of the class of faults that may occur with this computer system. The first fault was a power supply failure, the second was a failure in the I/O organisation of the central processor, the third was a failure in the I/O buffer, and the fourth was a failure in the I/O bus.

IV. 4. Data recording

Subjects were required to think aloud during the problem solving sessions, that were recorded on tape through a tape recorder attached to the subject and analyzed later. The subjects received some instruction on thinking aloud before the actual experiment started. Apart from the collection of these concurrent verbal protocols, various other measures were used, such as the number of problems solved to completion, the results of the knowledge test, a judgment made by the instructor immediately after each problem, a judgment (blind) about the quality of the solution, the interaction with the guidance system, and the interaction with the computer system itself.

IV. 5. Procedure

For those conditions in which the subject was guided by a system, a half-day was spent on familiarizing the subject with the guidance system. This turned out to be sufficient, as measured by a test of system understanding. Before starting the experiment, a knowledge test was administered. This knowledge test was designed to tap a representative part of the declarative knowledge that is needed to solve the problems in this experiment. The knowledge test has been validated with expert subjects, i.e., personnel that the Navy has assigned the task of teaching theory or practice in this particular computer system. The knowledge test provides a basis for what students know declaratively about the computer system to diagnose, and is especially important as basis in the no-system condition since it will be rather difficult to apply the correct strategy if one lacks the necessary domain knowledge. After the knowledge test was administered,
the first fault had to be found. Subjects were given a maximum of one hour to find each fault. There was a small 15 minutes break between the faults, with a longer lunch break between the second and third fault. Depending on the condition, the experiment lasted either 1 or 1.5 day.

IV. 6. Results

IV. 6. A. Quantitative results

Knowledge tests were scored anonymously by one rater. One point could be scored for each question, so the maximum score obtainable was 21.

Protocols were transcribed literally and coded according to the following coding scheme with four categories (a simplified version of the more elaborate coding scheme developed by Schaafstal, 1991):

1 / problem description (P), e.g., “led on A26 burns”;
2 / hypothesis generation (H), e.g., “fault in translator”;
3 / hypothesis testing (T), e.g., “measurement with voltmeter”;
4 / conclusion (C), e.g., “5V, high, so normal value”.

A sequence of hypothesis generation, testing and conclusion (H-T-C) was coded with a number (e.g., H1-T1-C1) if the testing and conclusion belonged to one hypothesis. In this way, the protocols were reduced to a number of H-T-C sequences, together with a short description of what hypothesis was tested and how. All protocols were coded in this way and were given to two independent raters (experts). Raters could not tell, on the basis of the coded protocols, what condition subjects were assigned to, so all protocols were scored blind to condition. Raters were asked to rate the solution quality using five categories: wrong (0), just beginning (.25), on the right track (.5), correct not tested (.75), correct and tested (1.0).

The spearman rank order correlation between the two raters was .76. This was considered sufficiently high to use the average of the two raters as the dependent variable.

Table II shows the proportion of four malfunctions correctly solved, using a lenient (> .50) and a strict (> .75) criterion.

<table>
<thead>
<tr>
<th></th>
<th>lenient criterion</th>
<th>strict criterion</th>
</tr>
</thead>
<tbody>
<tr>
<td>control group</td>
<td>.50</td>
<td>.25</td>
</tr>
<tr>
<td>knowledge only group</td>
<td>.46</td>
<td>.29</td>
</tr>
<tr>
<td>task structure group</td>
<td>.25</td>
<td>.12</td>
</tr>
<tr>
<td>Average (across groups)</td>
<td>.40</td>
<td>.22</td>
</tr>
</tbody>
</table>
Training of systematic diagnosis

Using this average and considering any scores higher than .5 as evidence for correctly solving the problem (a lenient criterion), the subjects solved on average 40% of the four malfunctions. Using a stricter criterion of .75, only 22% of the malfunctions was solved. The control group and the domain knowledge group each solved 50% (25% and 29%, respectively, using a strict criterion), whereas the task structure group solved only 25% (12% using a strict criterion) of the four malfunctions presented.

The main question of interest was whether subjects' solution quality was higher for the computer-supported groups than for the control group, and whether the "task structure plus domain knowledge" group performed better than the "domain knowledge alone" group. As it turned out, the last two malfunctions (faults 3 and 4) were more difficult than the first two malfunctions (baseline faults 1 and 2). All three groups therefore showed, on average, worse performance on the last two malfunctions. Taking into account initial baseline differences between groups, performance decrements were calculated for each group. Table III shows the solution quality for the three groups, averaged for the first two (baseline) and the last two (experimental) faults.

Table III
Solution quality for the control group, the domain knowledge only support group, and the domain knowledge plus task structure support group

<table>
<thead>
<tr>
<th>Group</th>
<th>Control</th>
<th>Knowledge only</th>
<th>Task structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault-type and number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>baseline (1+2)</td>
<td>.63</td>
<td>.59</td>
<td>.48</td>
</tr>
<tr>
<td>experimental (3+4)</td>
<td>.37</td>
<td>.40</td>
<td>.31</td>
</tr>
<tr>
<td>Decrement</td>
<td>.26</td>
<td>.19</td>
<td>.17</td>
</tr>
</tbody>
</table>

Averaged across groups, the decrement between the baseline and the experimental faults was statistically highly significant, F(1,17)=10.90, p=.004. Hence subjects performed worse on the last two than on the first two faults. A breakdown by group was performed in order to find out whether the decrement differed among groups. The control group showed the largest decrement, .26 (from .63 to .37), the domain knowledge group showed a decrement of .19 (from .59 to .40), and the task structure group showed a decrement of .17 (from .48 to .31). A repeated-measures multivariate analysis of variance, with average performance on the first two malfunctions as a covariate, did not show a main effect of group, F(4,20)<1, while showing a marginally significant effect of the covariate, F(2,10)=3.68, p=.06. Without the covariate, the three groups also did not differ from each other, F(2,15)<1.
A Tukey HSD multiple comparison between the control group and the two experimental groups showed no significant differences, all \( p's > .10 \). Therefore, when taking into account performance on the first two malfunctions, the three groups did not differ significantly from each other.

A second question of interest was whether performance on the knowledge test was predictive of fault-finding performance. The correlation between the total score on the knowledge test and the average score on the four malfunctions was \( .27 \) (\( p > .10 \)). Therefore, having theoretical knowledge relevant for diagnosing the four malfunctions in this experiment was barely related to actual fault-finding performance. On the other hand, lacking this knowledge on a pre-test was not a barrier to successful fault-finding performance, as witnessed by the three best-performing subjects who all scored well below the average on the knowledge test (average for the three subjects: 8.9 or 42% of the total possible; overall average: 11.5 or 55%).

In order to understand more precisely what subjects were doing when they were fault-finding, we took a closer look at the coded protocols. We expected the computer-supported groups to be more structured in their problem solving, partly because the immediate availability of relevant documentation would lead to a lower memory load, a smaller chance of forgetting one’s hypothesis being tested, and a larger number of hypotheses that can be generated. We therefore counted the number of Hypothesis-Test-Conclusion sequences for each subject. As a measure of “structuredness”, only H-T-C sequences belonging to one hypothesis were counted. Table IV shows the average number of H-T-C sequences for the three groups.

### Table IV

<table>
<thead>
<tr>
<th>Group</th>
<th>Control</th>
<th>Knowledge only</th>
<th>Task structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault-type and number</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>baseline (1+2)</td>
<td>2.55</td>
<td>2.25</td>
<td>2.00</td>
</tr>
<tr>
<td>experimental (3+4)</td>
<td>3.30</td>
<td>2.70</td>
<td>1.95</td>
</tr>
<tr>
<td>Increment</td>
<td>0.75</td>
<td>0.45</td>
<td>-0.05</td>
</tr>
</tbody>
</table>

Contrary to what we expected, the number of H-T-C sequences increased the most for the control group (from 2.55 on the first two malfunctions to 3.30 to the last two malfunctions, an increase of .75), and slightly less for the domain knowledge group (from 2.25 to 2.70, increase of .45). For the task structure group, the number of H-T-C sequences actually decreased from 2.0 to 1.95. Closer inspection of the log files...
Training of systematic diagnosis

revealed the reason behind this unexpected result: subjects in the task structure group often listed a large number of hypotheses in the beginning and then went on testing these without explicitly mentioning them again. A more appropriate measure would therefore be to focus only on the T-C sequences, since all groups would have to test their hypotheses, no matter when they were generated. Table V shows the average number of T-C sequences for the three groups.

**Table V**

*Average number of Test-Conclusion (T-C) sequences for the control group, the domain knowledge only support group, and the domain knowledge plus task structure support group*

Nombre moyen de séquences « Test-Conclusion » (H-T-C) pour le groupe-contrôle, le groupe avec soutien fondé sur les connaissances du domaine seules et le groupe avec soutien fondé sur les connaissances du domaine et la procédure

<table>
<thead>
<tr>
<th>Group</th>
<th>Control</th>
<th>Knowledge only</th>
<th>Task structure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault-type and number baseline (1+2)</td>
<td>2.90</td>
<td>2.50</td>
<td>2.25</td>
</tr>
<tr>
<td>experimental (3+4)</td>
<td>3.65</td>
<td>3.50</td>
<td>2.90</td>
</tr>
<tr>
<td>Increment</td>
<td>0.75</td>
<td>1.00</td>
<td>0.65</td>
</tr>
</tbody>
</table>

Counting T-C sequences showed an increase of .75 for the control group, of 1.0 for the domain knowledge group, and of .65 for the task structure group. Tukey's HSD pairwise comparisons showed no significant differences among the groups, all p's>.10.

IV.6.B. Qualitative results

During the experiment notes were taken of the mistakes made and difficulties encountered by subjects. Together with the verbal protocols, these data give a rich picture of the problems encountered by novice troubleshooters:

1 / Inappropriate use of measuring instruments and tools, for example: wrong setting of oscilloscope, not knowing what a signal means, wrong operation and read-out of Remote Control Panel, measuring while computer is in incorrect state, and incorrect counting of measurement points.

2 / Incorrect use of documentation, for example: inaccurate reading of Fault Finding Procedures, unable to find information, and unfamiliarity with abbreviations.

3 / Incorrect execution of safety procedures.

4 / Misunderstanding of negative logic, order of signals, and function of circuits.
Incorrect use of the documentation was less problematic when system support was given. This was one of the aims of both versions of the computer support system, since both versions contained, in electronic format, important and easy-to-find system documentation. The other problems encountered by our subjects were the same in each condition. What is surprising in these results is the lack of basic measurement skills. Obviously, even when one's reasoning is highly structured, a fault will not be found if one lacks the skills to test one's hypotheses.

V. DISCUSSION

The main finding of practical significance of the studies carried out is the relatively poor troubleshooting performance displayed by military personnel who have just completed a course and passed their exam. Across 18 subjects, only 22% of the malfunctions (40% using a more lenient criterion) was diagnosed correctly. Moreover, a score of 11.5 out of a possible 21 points (55%) was obtained on a knowledge test. There may be several explanations for this rather low performance level. First, the course completed was the first course taken on this subject, and the subjects had not had opportunities for extensive practice in troubleshooting on board of frigates. Second, the problems posed to them in the experiment were considered to be quite difficult.

An attempt to support these subjects by different types of computer guidance resulted in somewhat better performance relative to a control group. However, the results failed to reach significance. Several factors may have contributed to this lack of a statistically significant effect.

First, we feel that the subjects in the present experiments encountered so many difficulties at a basic level, that any support at higher levels, such as that of a task structure, is premature. We observed many problems with basics such as oscilloscope handling, accurate reading of fault isolation procedures and basic system knowledge. Structured thinking is made virtually impossible when subjects encounter one impasse after another. Possession of basic troubleshooting skills is required before any higher-order skills can be taught successfully. We frequently observed subjects in the task structure condition to use the computer as an administrative system rather than as an aid to guide their troubleshooting. Some subjects spent so much time with basics that they completely forgot about the computer aid.

Second, the disappointing performance by the task structure group may be due to the relative unfamiliarity of the task structure and hypothesis tree concepts and the poorer base-line performance (in terms of number of malfunctions correctly solved) of the task structure group. In order to really change subjects' approach to troubleshooting, far more practice is necessary than the three hours of practice subjects now received. The poor baseline performance was taken into account statistically, but may nevertheless have been an indication of a lower level of ability of this particular group, due to unfortunate sampling. This lower level of ability may have prevented this
group from fully grasping the relatively complex and abstract concepts of task structure and hypothesis tree.

Despite the lack of statistically significant results as far as the different forms of computer support are concerned, this research has nevertheless yielded many insights into real-life troubleshooting. For instance, the lack of a substantial correlation between theoretical knowledge and troubleshooting performance casts doubts on the heavy emphasis placed on theory in training courses. On the basis of our results, we would favor a heavier emphasis on training basic skills in practice. At the same time, theoretical education should be focused more on the functions of higher-level components than on the way these functions are realized physically. We expect trainees to respond more favorably to task level support once these changes in their practical and theoretical curriculum have been implemented.

We have implemented the ideas outlined above in a add-on mini-course that is given after the regular course in the computer system that was the object of our study. The mini-course lasts for five days, thus giving students much more opportunity to practice the structured approach to troubleshooting. In order to make the task structure more concrete, we have developed a “troubleshooting form” that guides students through each malfunction in a structured way. In our mini-course there is a heavy emphasis on the practice of troubleshooting itself: students are required to find 16 faults during the five-day course. Moreover, basic measurement skills such as knowing what signal to expect and oscilloscope handling are practiced extensively.

We have also concluded that practice with the task structure by itself is of limited value unless it is supported by relevant functional system knowledge. A systematic approach to troubleshooting and functional system knowledge should go hand in hand when training for successful troubleshooting. This is because the systematic approach can only be maintained when large functional blocks can be eliminated and the troubleshooter can gradually expand the remaining functional blocks. In collaboration with an expert, we have made descriptions of the computer system at several levels of abstraction. This functional description is taught jointly with a systematic approach to students in the mini-course.

We are currently evaluating this mini-course by presenting the same four malfunctions as used in the experiment discussed above to students who have taken this extra course. Preliminary results are very promising in that the number of faults successfully solved has at least doubled. Although we have made a slight detour in turning our attention away from the knowledge-based support tool and towards direct educational interventions, once these interventions prove promising enough we may return to the computer-based support tool and improve it based on our newly acquired knowledge.
Acknowledgements

We would like to thank Ltzei J. Bisschops, klTZE Ir. W. W. Schalkoort, SMJR J. Hoekman, and SGTWE J. Sackman of the Royal Netherlands Navy for generously providing domain expertise and their support in all phases of the project.

REFERENCES


Training of systematic diagnosis

(Ed.), Advances in Man-Machine Systems Research (pp. 139-193). Greenwich, CT: JAI Press.


SUMMARY

Two studies were carried out with the aim of developing better training methods for Naval maintenance technicians. The first study showed that novices experienced difficulties in two areas: 1) accessing relevant domain knowledge in particular problem solving phases (e.g., generating causes, testing causes); 2) keeping track of where they are in the process of diagnosis and deciding what to do next (strategic knowledge). The second study consisted of a more formal evaluation of these ideas, implemented in a knowledge-based guidance system. Results showed that support at the strategic level was not effective, possibly due to a lack of proficiency of basic skills in troubleshooting.

Key words: Diagnosis, Training, Cognitive task analysis, Knowledge technology.


Paper accepted in revised form: July 1995.