ABSTRACT
This paper describes our design and implementation of an expert system for computer performance evaluation (CPE). We explain objectives in designing an expert system for CPE, explain why a specific expert system development shell was selected, describe the structure of the expert system that resulted from the design, and provide examples of the rules comprising the expert system. An operational version of the expert system implemented for IBM’s MVS/370 and MVS/XA is briefly described.

INTRODUCTION
An expert system is a computer program that emulates the way that people solve problems. Like a human expert, an expert system gives advice by using its own store of knowledge that relates to a particular area of expertise. In expert systems terminology, the knowledge generally is contained in a knowledge base and the area of expertise is referred to as a knowledge domain. The expert system’s knowledge often is composed of both (1) facts (or conditions under which facts are applicable) and (2) heuristics (i.e., “rules of thumb”).

With most expert systems, the knowledge is stored in “IF/THEN” rules that describe the circumstances under which knowledge is applicable. These expert systems usually have increasingly complex rules or groups of rules that describe the conditions under which diagnosis or conclusions can be reached. Such systems are referred to as “rule-based” expert systems.

Expert systems are used today in a wide variety of fields. These range from medical diagnosis (e.g., MYCIN) to geological exploration (e.g., PROSPECTOR), to speech understanding systems (e.g., BABSAY-II), to laboratory instruction (e.g., SOPHIE). In 1987, Wolfgram, et al., listed over 200 categories of expert system applications, with examples of existing expert systems in each category. These same authors estimate that by 1995, the expert system field will be an industry of over $9.5 billion.

Over the past several years, numerous papers have been presented at the the Computer Measurement Group (CMG) International Conferences describing the design and use of expert systems in the computer performance evaluation area. Additionally, several expert systems for computer performance evaluation have become commercially available (e.g., CA-ISS THREE, MINDOVER/MVS, the DASD Advisor, and MVS Advisor). Clearly, expert systems technology is playing an increasing role in computer performance evaluation.

We decided to explore the use of expert systems technology in our business (Computer Management Sciences, Inc. is a consulting firm specializing in the computer capacity management field.). We examined the expert systems commercially available in the CPE field. However, none of those met our needs. In 1987, we began an effort to design our own expert system for CPE.

OBJECTIVES IN DESIGNING AN EXPERT SYSTEM FOR CPE
The following summarizes our objectives in designing an expert system for computer performance evaluation:

- We wanted to perform projects with less experienced personnel and thus less cost to our clients.
- We wanted to have consistent analysis of data.
- We wanted to capture the analysis process, so we don’t “reinvent the wheel” in our projects.
- We wanted to make sure that senior analyst/consultant expertise is available in case the senior analyst or consultant is unavailable.
- We wanted to train our junior staff in CPE techniques.

EXPERT SYSTEM DEVELOPMENT SHELLS
Expert systems can be developed using any suitable computer language (e.g., BASIC, C, etc.). However, researchers quickly realized that using a development shell would significantly hasten the process of developing an expert system in a particular knowledge domain. A development shell contains the control mechanism to interpret the rules, direct flow between rules, provide working memory, and serve as an
interface between user of the expert system and the rules.

In 1987, Wolfgram, et al., listed 60 expert system development shells or languages that were being used to develop expert systems. The prices of these expert system development shells ranged from inexpensive (e.g., less than $100) to expensive (e.g., over $50,000).

SELECTING AN EXPERT SYSTEM DEVELOPMENT SHELL

We wished to develop an expert system that we could use on personal computers (PCs), so we restricted our examination of expert system shells to those that operate on personal computers. We wanted the expert system shell to possess the following design characteristics:

- The development shell should be easy to use from a development view, an enhancement view, and an end user view. When presenting their paper, Domanski and Ingrassi described the inordinate amount of time that they spent in learning to use the PROLOG expert system development shell. We wanted to spend our time capturing the expertise, rather than learning how to use the tool. After examining PROLOG, we decided that it was unsuitable for our purposes.

- The development shell should provide an easy-to-modify knowledge base. We wanted to have a system that would become operational quickly, and could be enhanced easily. This meant that the knowledge base should be easy to understand and easy to modify.

- The development shell should provide for the uncertainty of analysis (i.e., experts aren't always "sure" of some analysis, but they can be "90% sure" or "almost sure"). The development shell should provide a means of assigning uncertainty (or certainty) values to its evaluations.

- We wanted a well-documented product. Some of the expert system shells that we evaluated were technically good, but lacked good documentation.

- The development shell should produce an expert system that would be quick to execute.

- The development shell should be inexpensive to buy and inexpensive to execute. We planned to equip our analysts with copies of the expert system loaded onto a laptop PC. Some of the development shells cost over $20,000; we didn't evaluate these.

- We planned to interface the expert system with a variety of other products (e.g., SAS·, queuing models, spreadsheets, etc.). The development shell should be modular and should provide a good interface to other languages (e.g., it should be able to call another language from within the expert system).

- The development shell should provide a good interface for I/O from the host system that we would be evaluating (to extract information from a host mainframe) and a good interface to users (so that users can both experiment and enter data).

- Since we expected the expert system to undergo continual maintenance and enhancement, we wanted the development shell to facilitate internal documentation that could be provided to users of the knowledge base.

There were other considerations that we used to evaluate the expert system development shells, but these were the main ones:

After examining a variety of expert system development shells, we chose VP-Expert15 as the development shell for our expert system. VP-Expert didn't meet all of our needs, but it met most of them.

- VP-Expert provides a language that is powerful and is extremely easy to learn. Within a short time (less than two weeks) after purchasing VP-Expert, we were executing modules of our CPE expert system.

- VP-Expert provides an easy-to-modify knowledge base, using a straightforward IF-THEN construct. The knowledge base/inference engine combination provides for both forward and backward chaining.

- VP-Expert provides "Confidence Factors" that can be associated with the output of a particular rule. A simple SORT function
allows the arranging of rule outputs so that the most likely output can be accessed first.

- Paperback Software, Inc. provides good documentation, including a regular newsletter for users.

- VP-Expert is inexpensive. While we paid $96.00 for our initial version, we have purchased additional copies from discount houses for less than $70.00. Considering the other features, we think that VP-Expert is an incredibly good bargain! (Note that Version 2 of VP-Expert is priced at $145 from discount houses—a price that still is very reasonable.)

- VP-Expert can directly execute calls from its inference engine (or even from its knowledge base) to DOS .COM files, .EXE files, or .SAT files. This facility can be used to access any DOS program.

- VP-Expert has excellent internal documentation, and provides a "BECAUSE" statement that can be used to communicate reasoning from the expert system developers to users consulting the expert system. Users thus can trace the reasoning chain of the expert system.

On the downside, VP-Expert is disappointing slow to execute. The execution speed is not bothersome when operating in Consulting mode with simple questions. However, it became frustrating when analyzing a large amount of information downloaded from a host computer system. We execute our expert system on a 20MHz 386 personal computer. However, the execution speed is marginally acceptable when analyzing large amounts of performance data from a host computer system.

On the downSide, VP-Expert is disappointingly slow to execute. The execution speed is not bothersome when operating in Consulting mode with simple questions. However, it became frustrating when analyzing a large amount of information downloaded from a host computer system. We execute our expert system on a 20MHz 386 personal computer. However, the execution speed is marginally acceptable when analyzing large amounts of performance data from a host computer system.

DESIGNING AND IMPLEMENTING THE EXPERT SYSTEM

The expert system that we designed is intended to analyze the performance of IBM computer systems operating under MVS/370 and MVS/XA operating systems. We combined expert systems technology with the experience that CMS staff members and our consultants have gained while analyzing hundreds of computer systems. We further (1) analyzed dozens of IBM manuals (and manuals by Amdahl and Nas); and (2) read numerous papers published in CMG Transactions and Proceedings, and in SHARE and GUIDE Proceedings. The result is a system that provides a detailed diagnosis of many of the constraints to improved computer system performance, and gives specific advice on how to eliminate the constraints.

The expert system consists of an inference engine and a variety of knowledge bases.

Inference Engine. The Inference Engine currently operates in one of two modes: Consulting Mode or Automatic Mode.

When our expert system is operating in Consulting Mode, a user extensively interacts with the system. The user provides detailed information describing the environment behavior, in response to questions from the system. (VP-Expert makes this design feature trivial to implement; by simply setting a parameter, you can specify that whenever the expert system shell doesn’t "know" something, it should ask for the information!) The expert system analyzes the information and provides interactive responses. In addition, the user may be provided references to source material which may be read in order to better understand the analysis and recommendations. The user can ask "why" to receive an explanation associated with each rule that fires, or the user can optionally use VP-Expert "RECORD" mode to produce a file of these explanations.

The Consulting Mode is an excellent vehicle for teaching our junior analysts (1) what information should be used to diagnose constraints to improved computer system performance, (2) where to obtain the information, (3) how to analyze the information, and (4) what recommendations could be made based upon the analysis. Additionally, the Consulting Mode is useful when analyzing low-volume data (e.g., data displayed by the CICS CSTT transaction, or printed with CICS Shutdown Statistics).

However, the Consulting Mode is of limited use when analyzing data collected over several weeks while performing a project for one of our clients. We quickly realized that the tedium of keying in data and responding to questions was unworkable in a normal project environment. Consequently, we implemented a different mode, in which the expert system analyzes information that is downloaded from the host computer system. We called this mode of operation Automatic Mode. When our expert system is operating in Automatic Mode, the system is essentially non-interactive. The expert system support software
executing on the host computer system obtains environment behavior information, and provides the information directly to the expert system. We use SAS executing on the host system and SAS/PC executing on our PC.

The SAS code executing on the host computer system reads and shapes performance data. We use the SAS/PC micro-to-host link (using PROC DOWNLOAD) to transmit information from the host to our personal computer. We normally use the Async connection download option, although SAS/PC micro-to-host supports other modes of downloading data (such as an ERMA board). VP-Expert does not processes SAS datasets. However, we use PROC DBF to convert the SAS datasets to dBASE files, which VP-Expert can import.

Our expert system analyzes the performance information, makes diagnoses, and gives tuning advice based upon this analysis. Our analysts can display the diagnoses and tuning advice on-line, or produce hardcopy output.

Knowledge Base. The Knowledge Base provides the essential knowledge which our expert system's inference engine uses to analyze computer performance. The Knowledge Base consists of the following individual bases:

- Rule Base
- Environment Description Base
- Guidance Base
- Diagnosis Base
- Advice Base
- Reference Base
- Environment Behavior Base

The Rule Base contains the rules used by the inference engine to analyze computer performance. The Rule Base provides the IF/THEN construct with which the other bases are evaluated by the inference engine. The Rule Base is modular; it is partitioned into:

- CPU Rules
- SSM Rules
- Memory Rules (Real, Virtual, and Auxiliary Storage)
- DASD I/O Rules
- Other I/O Rules
- Application Rules (e.g., CICS rules, TSO rules, S2K rules, etc.)

Some rules in the Rule Base contain more than simple IF/THEN statements. These rules significantly expand the IF/THEN construct, performing detailed calculations on the data, much as a human expert would. Other rules contain M/M/1 queuing models that supplement the guidance contained in the Guidance Base. (For example, the M/M/1 queuing models can predict the expected I/O response time, considering the level of I/O activity to a particular DASD. The expert system compares the expected I/O response time to the actual I/O response time so that it can make "good" or "bad" value judgments about the I/O response time. This process is similar to that performed by a human expert.)

The result of these calculations or models directly effect the recommendations that the expert system makes, and effect the confidence with which the recommendations are made.

The initial Rule Base had few (less than 200) rules. However, our modular design allows us to add rules as they are devised. The Rule Base is continually undergoing enhancement. We plan to update the Rule Base after each project that we perform.

The Environment Description Base describes the environment. The Environment Description Base contains information describing the hardware characteristics (CPU types, memory size and type, DASD types and configuration, shared components in a multiple system environment, etc.), operating system parameters implemented by the installation (usually extracted from the IEAOPT, IEAICS, and IEAIPS members of SYS1.PARMLIB), and unique file placements (the VOLSER of system packs, paging packs, TSO packs, etc.).

The Environment Description Base is created when our analysts first begin the consulting engagement. This is the normal information that is acquired when we execute the "Understand Environment" task that is performed at the beginning of one of our projects. The Environment Description Base is modified when changes are made to the environment being analyzed.

At present, the Environment Description Base must be manually created. However, we plan to create most of the labor intensive parts automatically (e.g., we plan to read SYS1.PARMLIB members and transfer appropriate information directly into the Information Base).

The Guidance Base contains "value judgement" information that guides our expert system in its analysis of the particular installation. The Guidance Base contains such information as (1) LOW/NORMAL/HIGH criteria to be applied against particular information contained in the Information Base, (2)
uncertainty values for each of the evaluation criteria, and (3) specific installation overall performance objectives that guide the expert system in assessing computer performance against management objectives.

The Guidance Base is flexible, to allow value judgments to be applied to both general and specific evaluation components. For example, a particular level of DASD activity might be generally acceptable for most volumes; a much lower level of activity might be desired for critical volumes. As another example, a particular level of paging might be considered acceptable for most applications. However, a CICS application would tolerate much lower paging rates before performance would become sharply degraded. The Guidance Base allows our analysts to guide the expert system in its analysis of these unique situations. The Guidance Base allows our analysts to specify value judgments about thresholds of "good" or "bad" system behavior.

The Guidance Base will be changed by our analysts as they better understand the environment that they are analyzing or in response to unique situations.

The Diagnosis Base contains the narrative associated with a particular conclusion that our expert system reaches based upon its analysis. These conclusions range from general ("I/O Response Time is too high") to specific ("There is too much seeking on VOLSER xxxxxx").

The Advice Base contains the narrative associated with a particular recommendation that our expert system makes based upon conclusions it has reached. The expert system associates "confidence levels" with the recommendations that it makes, sorts the recommendations into descending order, and presents the most likely recommendations. (A system parameter controls the confidence level below which the expert system stops making recommendations.)

The Reference Base contains references that relate specific rules, diagnoses, or advice to individual source documents. The references describe source documents from which we obtained the information upon which the rules, diagnoses, or advice are based.

The Environment Behavior Base contains information describing how the computer environment behaves during the interval that our expert system is to analyze. The Environment Behavior Base contains such information as: utilization of components, queuing for resources, response and turnaround times of workload, etc.

When our expert system is operating in Consulting Mode, the Environment Behavior Base is created by our analysts (often in response to direct questions from the expert system). When the system is operating in Automatic Mode, the SAS support software in the host system extracts environment behavior information from such sources as SMF data, RMF data, CICS Monitoring Facility data, System 2000 Diagnostic Log data, MICS data base, SAS/MKG data base, etc. As described earlier, the information is then downloaded to SAS/PC executing on the personal computer.

EXAMPLE

Our example project is an IBM environment with over 100 IBM 3380 DASD. The system is operating under MVS/XA, and supports TSO, CICS, System 2000, and batch workload. We collected SMF Type 70 (series) records, CMF records (Performance Class and Exception Class), and certain System 2000 Diagnostic Log records.

Exhibit 1 shows the overall flow of data. We collected the data using the tools just mentioned. We then shaped the data using SAS on the host computer system and placed the data into a performance data base. We dialed the host computer from our PC, using standard PC communications software (we have used both Crosstalk and ProComm Plus for this purpose; both packages allow you to exit from the communications software after establishing logon to the host). Using the SAS/PC host-to-micro link, we downloaded selected data from the performance data base (executing PROC DOWNLOAD via REMOTE SUBMIT).

With our current implementation of the expert system, we download data in response to specific rules that we wish to execute (e.g., we download CICS data if we are executing the CICS rules, or DASD data if we are executing the DASD rules). We currently must operate with manual intervention, selecting the specific knowledge base rules that we wish to execute, and downloading the associated data. However, we plan to enhance our system to eliminate this manual intervention.
Exhibit 2 contains a partial list of the data that would be downloaded if we were executing the CICS knowledge base.

- Average response time for "fast" transactions
- Number of CICS limit conditions reached
- Number of CICS program compressions
- Number of CICS page in operations
- CICS transaction wait for VSAM buffer

Exhibit 3 lists a sample rule that analyzes the CICS data, the diagnosis that results from the rule firing, the advice that is generated based on the diagnosis, and the reference that is associated with the rule. (Note that the VP-Expert coding has been slightly altered to make the example more easily understood.)

The "CNF = PAGEIN/10" contained in the rule coding represents the confidence with which the rule is true. (The values 5 and 10 were arbitrarily selected for this example, based upon the heuristic that paging should be no greater than 5-10 page in operations per second.) The rule fires if the paging rate is greater than 5 per second, and paging will be a problem with increasing confidence as the paging rate approaches (or exceeds) 10 pages per second. VP-Expert will automatically associate the computed confidence interval with the rule's result. (The rule's result is shown as in Exhibit 3 as "PROBLEM = PAGE IN OPERATIONS").

\[
\text{IF PAGEIN GT 5/SECOND}
\]
\[
\text{THEN PROBLEM = PAGE IN OPERATIONS}
\]
\[
\text{CNF PAGEIN/10}
\]

\textbf{DIAGNOSIS: Excessive CICS page in operations.}

\textbf{ADVICE:}
1. Implement memory fencing or increase the minimum working set size in PWSS.
2. Control CICS throughput using MXT, AMXT, CMXT, and DLTHRED.
3. Review other possible actions listed on page 186 of the CICS Performance Guide.

\textbf{REFERENCE: CICS Performance Guide, pages 158-159.}

Exhibit 4 lists another sample rule that analyzes the CICS data, the diagnosis that results from the rule firing, the advice that is generated based on the diagnosis, and the reference that is associated with the rule.

As described earlier, the CNF statement contained in the rule coding in Exhibit 4 computes the confidence with which the rule is true. (The values 4 and 12 were arbitrarily selected for this
IF PROGCOMP GT 4/HOUR
THEN PROBLEM = PROGRAM COMPRESSION
CNF PROGCOMP/12

DIAGNOSIS: Program compression occurs too frequently.
ADVICE: 1. Increase CICS region size.
2. Control CICS throughput using AMXT, MXT,
GAXT, or DLTHRED.
3. Review other possible actions listed on page
187 of the CICS Performance Guide.

Sample Expert System Analysis
EXHIBIT 4

example, based upon the heuristic that
program compressions may be a problem
if they occur more often than once
every 15 minutes, and definitely are a
problem if they occur more often than
once every 5 minutes. The actual
values in our expert system would be
extracted from our Guidance Base. The
values in the Guidance Base would vary
depending upon a variety of factors,
such as whether the CICS program
libraries are located on cached DASD, etc.

VP-Expert automatically associates the
confidence factor with which a rule
fires with the rule output. Consequently, these outputs can be
analyzed by other rules to select the
most likely cause of a performance
problem. VP-Expert allows variables to
have multiple values represented as a
push-down stack. Each value retains
its confidence factor. In the above
examples, the variable "PROBLEM" can
simultaneously represent a number of
rule outputs, with their associated
confidence factors. The multiple values
can be sorted based upon the confidence
factors, and individual values can be
POPed off the stack. This feature
greatly facilitates ordering and
analyzing rule outputs.

CONCLUSION

We wanted to design an expert system to
capture the expertise that had been
gained by experience in the computer
performance evaluation field. We
intended to capture most of the
expertise, but we did not expect to be
able to analyze every problem.
Further, we did not hope to encode the
"jumps in logic" characteristic that is
one of the factors of an expert. Thus,
we did not intend for our CPR expert
system to replace a CPR expert.

We felt that our expert system should
be quick to implement and easy to
modify. Our goal was to get something
operational, and assess its benefits.
We could then enhance the system as
needed.

From these perspectives, we have met
our objective. We have designed and
implemented an expert system for
computer performance evaluation. We do
not claim that the system will
correctly analyze every situation. We
do believe that it will provide
excellent guidance for many of the
performance situations that are
encountered in a typical MVS
environment. The system has proven to
be extremely useful in the projects in
which it has been employed.

Capturing this expertise aids "experts"
in that (1) they analyze systems
performance in a consistent fashion,
(2) they do not have to remember every
every analysis area, and (3) they can
focus on developing new analysis
techniques or conclusions. It aids our
more junior analysts in that (1) they
can determine what information to
analyze, (2) they learn the analysis
techniques that are used by experts,
and (3) they can study the logic paths
to see why particular decisions were
made.

We plan to continue enhancing the
expert system that we have designed.
As each project is completed, the
analysis process will be formally
reviewed and, where possible, reduced
to the expert system rule construct.

1Buchanan, B.G. and Shortliffe, E.H.
Rule-based expert systems: The MYCIN
experiments of the Heuristic
Programming Project, Reading, MA.
1983.

2Duda, R.O.; Barrett, J.; Gaschnig, K.;
Hart, P.E.; Konolige, K.; Regoh, R.;
and Elocum, J. Development of the
PROSPECTOR consultation system for
mineral exploration. Final report.
SRI Projects 5821 and 6415,
Artificial Intelligence Center, SRI

3Erman, L.D.; Fickas, S.F.; and London,
P.E. The design and an example use


DASD Advisor, Boole & Babbage, Inc. 510 Oakmead Parkway, Sunnyvale, CA 94086.

MVS Advisor, Domanski Sciences, 16 Colonial Court, Howell, NJ 07731.


SAS is the registered trademark of SAS Institute Inc., Cary, NC, USA.

VPEXPERT is a trademark of Paperback Software International, 2830 Ninth Street, Berkeley, CA 94710.

AUTHOR INFORMATION:

Donald R. Deese
Computer Management Sciences, Inc.
6076-D Franconia Road
Alexandria, VA 22310
(703) 922-7027