1987

A Domain Specific Expert System Tool for Building Photofinishing Process Diagnostic Systems

King Choi

Rochester Institute of Technology

School of Computer Science and Technology

A Domain Specific Expert System Tool for Building Photofinishing Process Diagnostic Systems

by

King F. Choi

A thesis submitted to

The Faculty of the School of Computer Science and Technology

in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science

Approved by: _____

Professor John A. Biles, RIT

_____

Dr. John Birkelund, Eastman Kodak

_____

Professor Peter Anderson, RIT

March 23, 1987

Revised sample statement for granting or denying permission to reproduce an RIT thesis.

Title of Thesis Photofinishing Process Diagnostics Expert System Tool

I King F. Choi hereby (grant, deny) permission to the Wallace Memorial Library, of R.I.T., to reproduce my thesis in whole or in part. Any reproduction will not be for commercial use or profit.

Or

I _____ prefer to be contacted each time a request for reproduction is made. I can be reached at the following address. _____

_____

Date March 25, 1987

Abstract

The knowledge-based method is a new and useful technique to build computer systems for applications which are heuristic rather than algorithmic in nature. This paper discusses how the approach can be applied to photofinishing process diagnostics. A prototype system for diagnosing Process E-6 for Kodak Ektachrome films is built to understand the knowledge structure and inferencing criteria for photofinishing process diagnostics. The experience gained from building the prototype Process E-6 diagnostic system has been applied to the design of an expert system tool for building photofinishing process diagnostic systems without the direct involvement of a knowledge engineer.

Contents

Chapter 1

Overview

The knowledge-based method is a new and useful technique to build computer systems for applications that are heuristic rather than algorithmic in nature. This paper discusses how this approach can be applied to photofinishing process diagnostics. A prototype system for diagnosing Process E-6 for Kodak Ektachrome films was built to understand the knowledge structure and inferencing criteria for photofinishing process diagnostics. The experience gained from building the prototype Process E-6 diagnostic system was then applied to the design of a domain specific expert system tool for building photofinishing process diagnostic systems. This domain specific expert system tool was developed in the VAX/VMS environment using mainly OPS5 and C.

Chapter two provides the background materials for understanding the subsequent chapters. It begins with an introduction to the photofinishing process and how it is controlled. It also contains a brief discussion and selected examples on expert systems and expert system building tools.

Chapter three describes the prototype Process E-6 photofinishing diagnostic system. This system was developed using OPS5 supplemented with FORTRAN routines. The knowledge structure and the inferencing criteria for diagnosing a photofinishing process are established based on this prototype.

Chapter four is the design of the domain specific expert system tool for building photofinishing process diagnostic systems. This tool consists of a knowledge base editor, a data definition module which pre-defines all the necessary frames, a shell main module which contains the structures for building up the target knowledge base, and an expert system skeleton module which merges with the target knowledge base to form the target expert system.

Chapter five discusses the implementation of this tool. It describes how the various components are implemented. The structure of the rules in a typical target knowledge base is also explained in this chapter.

Chapter six contains the concluding remarks for this project. The strength and the weaknesses of this domain specific expert system tool are discussed. Suggestions are made as to how this tool can be further improved.

Chapter 2

Introduction

2.1 Photofinishing Process

A photofinishing process is a procedure for converting a latent image on a photographic plate to an actual image.

It has been known for a long time that some silver salts, such as silver bromide and silver iodide (collectively called silver halides), are sensitive to light[Glaf 58,John 63]. When silver halide is exposed to light, the structure of the silver halide will be changed. It was also noted that unexposed and exposed silver halides have different chemical characteristics.

When a thin layer of silver halide emulsion is put on a plate which is later exposed to different light intensities, a latent image is formed on the plate. By placing the plate in a developer solution, the exposed silver halide is reduced to metallic silver while the unexposed silver halide is not affected. The unexposed silver halide can then be dissolved and washed away in the subsequent steps, leaving the metallic silver behind. Because of the special molecular structure of the reduced metallic silver, it actually appears in black rather than "silver" color, resulting in a black-and-white picture[Glaf 58,John 63].

Color processing[Bran 83,Spen 75] is more complicated. Based on the tri-color theory (red, green and blue for the additive method or their corresponding complementary colors, cyan, magenta and yellow, for the subtractive method), the whole color spectrum can

be reproduced by just three primary colors. In photography, the subtractive method is extensively used. Three layers of silver halide emulsion that contain (or are later caused to contain) dyes of cyan, magenta and yellow are made to become sensitive only to red, green, and blue light, respectively. Thus, three overlapping images of cyan, magenta and yellow colors are formed, resulting in a color picture.

There are many different photofinishing processes in use to-day[EK 80]. For example, the pictures that we take using Kodacolor films (or equivalent ones) are first processed using Process C-41 to develop the color negatives. The color negatives are then used with a printer to produce latent images on Kodak Ektacolor papers which are then developed using Process EP-2 to produce the color prints.

Color slides such as Kodak Ektachrome films are developed using Process E-6. Process E-6 is a reversal process which first develops the exposed film using a black-and-white developer. The film is then re-exposed and developed using a color developer resulting in a positive image.

Process K-14 is used for developing color slides on Kodachrome films. It includes a black-and-white developer followed by re-exposures and three color developers (one for each subtractive primary color) to form the positive image. It is by far the most complex photofinishing process.

2.2 Control of Photofinishing Process

Photofinishing is a chemical process that can be characterized by a set of state variables, or the chemical and physical parameters. A real-time direct control of a chemical process could be implemented if thorough feedback information of all state variables can be obtained at any time. Physical parameters, such as the temperature of a solution, the amount of time that a film is in a solution, the replenishment rate of a solution, etc., can be measured on a timely basis. However, chemical parameters, such as the composi-

tion and concentrations of chemicals in a processing solution, are very difficult to measure instantaneously and precisely. It seems that real-time direct control of a photofinishing process is beyond the technology available today[Thom 73,Jack 76].

Since photofinishing processes are dynamic in nature, the controlling parameters may drift with time with various degrees of severity. Control is further complicated by human factors such as mixing the chemicals, setting the temperatures, etc. A process that once produced excellent pictures may produce pictures that are no longer acceptable.

Thus, some way to keep a photofinishing process in control is necessary and this is usually done by human experts through examining the pictures produced. To do this objectively, a film strip (commonly known as a control strip) that is pre-exposed to different intensities of light under controlled conditions is processed with customers' work to determine the status of the process.

Figure 2.1 is a typical control strip for Process E-6. At the top of this control strip, there are three color patches (yellow, magenta, and cyan) which are pre-exposed only to blue, green, and red light, respectively. These patches provide an easy eye-examination of the quality of the individual color. In the middle of the strip is the "grey" scale. It is a series of patches exposed to increasing intensities of white light. Based on these patches, characteristic curves such as the one in Figure 2.2 can be drawn.

Usually, characteristic curves are characterized by four points corresponding to four levels (steps) of exposure. They are the minimum density step (D-min), low density step (LD or speed step), high density step (HD or color step) and maximum density step (D-max). The red, green, and blue densities at these exposure levels are read using a densitometer and compared against the reference values. The deviations from the reference values are then plotted to monitor the status of the process.

Figure 2.3 shows typical deviation plots on a Kodak color process record form no. Y-55.

Figure 2.1: A Process E-6 control strip

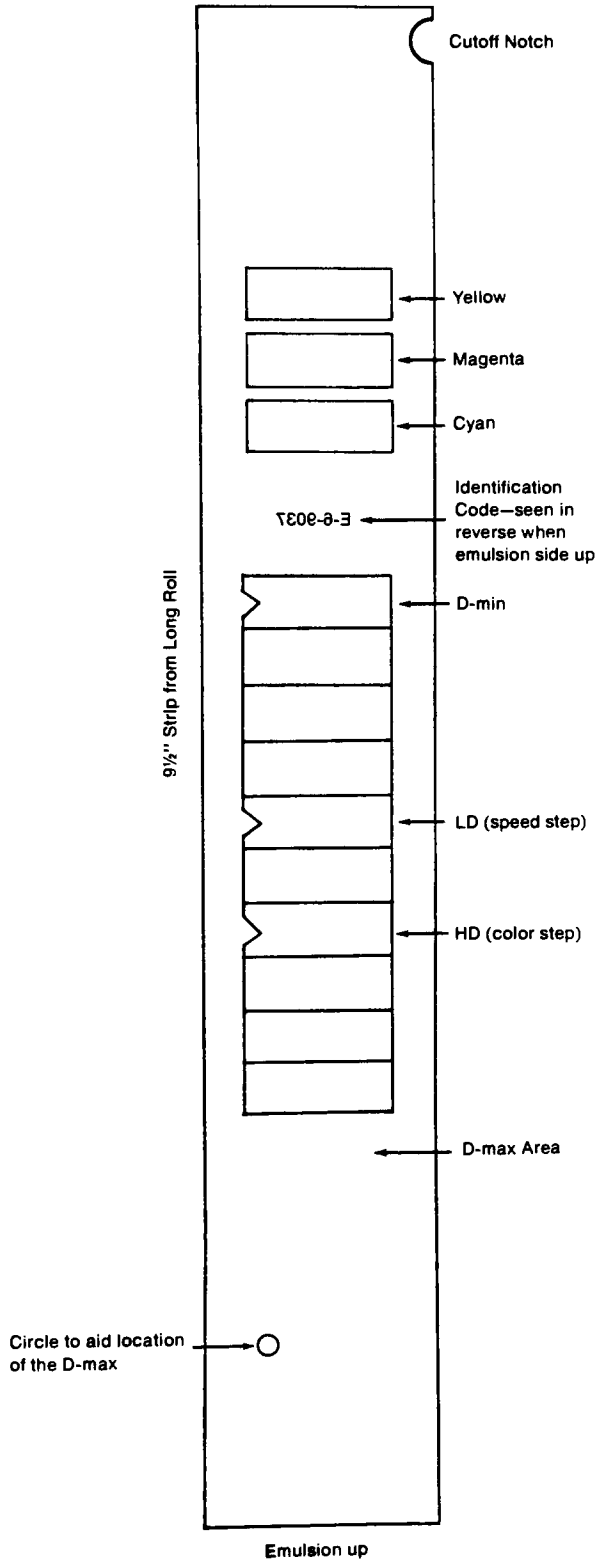
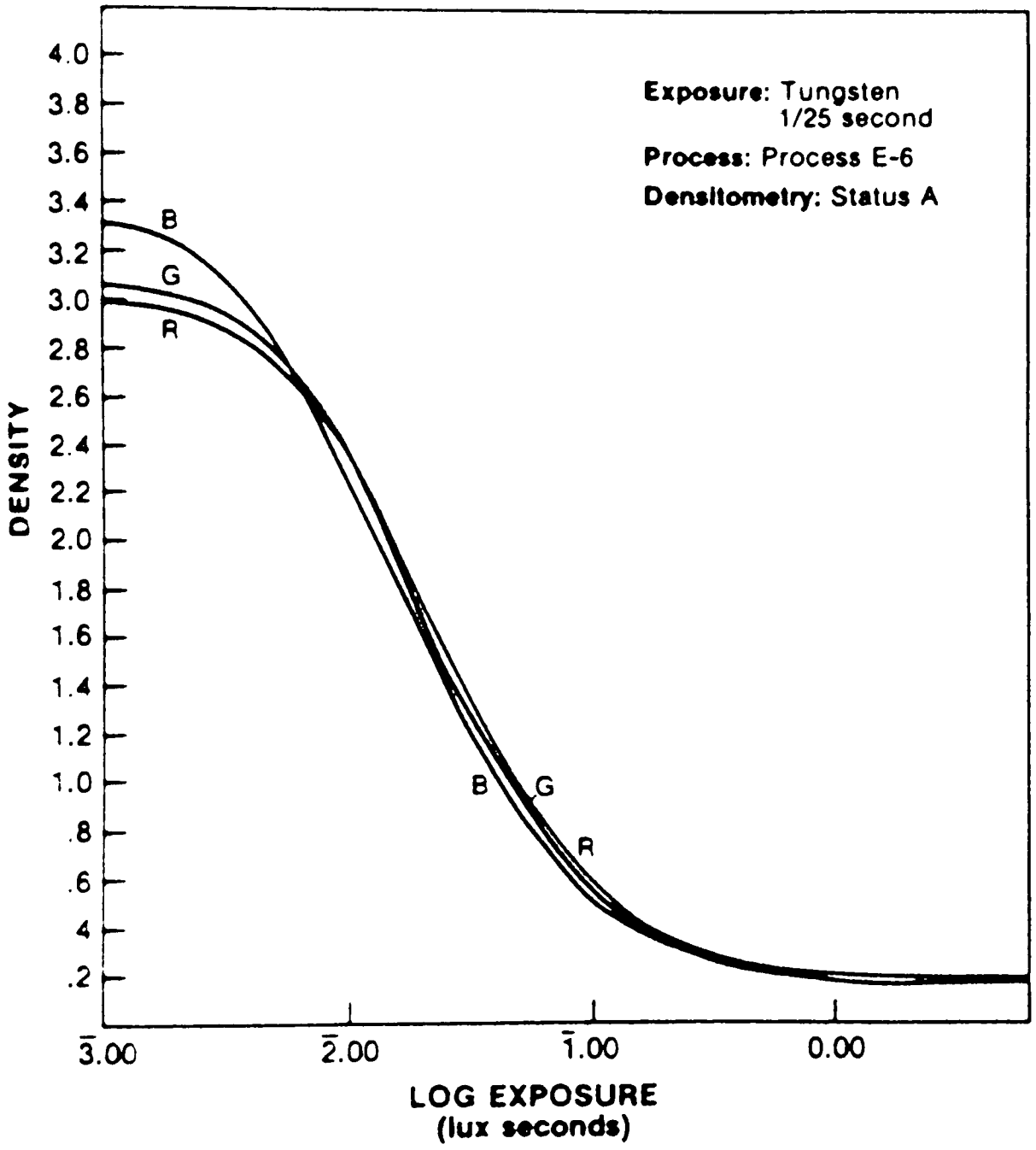


Figure 2.2: Characteristic curves of Kodak Ektachrome 160 professional film (tungsten)



In this figure, there are a number of solid lines and dotted line-pairs running across the form. The solid lines indicate the aims of the parameters. The closer the plots are to the aims, the better the pictures. For each pair of dotted lines, the one that is closer to the aim defines the action limit. When a parameter moves beyond the action limit, this implies that the process is beginning to get out-of-control and action should be taken to bring the parameter back. The dotted line that is further away from the aim defines the control limit. When a parameter goes outside the control limit, the process is totally out-of-control and processing should be stopped or unsatisfactory pictures will result.

Using Figure 2.3 as an example, when the blue speed step plotted on this chart moved above the action limit at 12:20 on September 5, it was discovered that the underlying cause for this problem was the first developer temperature too low. Corrective action was taken to adjust the first developer temperature and the process was brought back under control.

This process of determining the underlying cause or causes is known as photofinishing process diagnosis. To assist processing experts (or novices) in diagnosing process control problems, many control plots are documented showing how each chemical or physical parameter affects the color densities at various exposure levels. Figure 2.4 is an example of such control plots and it shows the effect of color developer overreplenishment.

Since there are many physical and chemical parameters, and each control plot only shows the effect of a single parameter, the control plots are only helpful in a limited way to determine the possible cause(s) when in real life, control problems usually have multiple causes. Over the years, a processing expert develops a set of hunches which are not documented in any way to solve processing control problems. However, these experts are hard to come by and many of them will sooner or later retire and carry with them their troubleshooting expertise.

Therefore, it appears helpful to have a computer-aided diagnostic system based on current expert system technology to preserve the process diagnostic expertise and to aid personnel

Figure 2.3: Process E-6 deviation plots

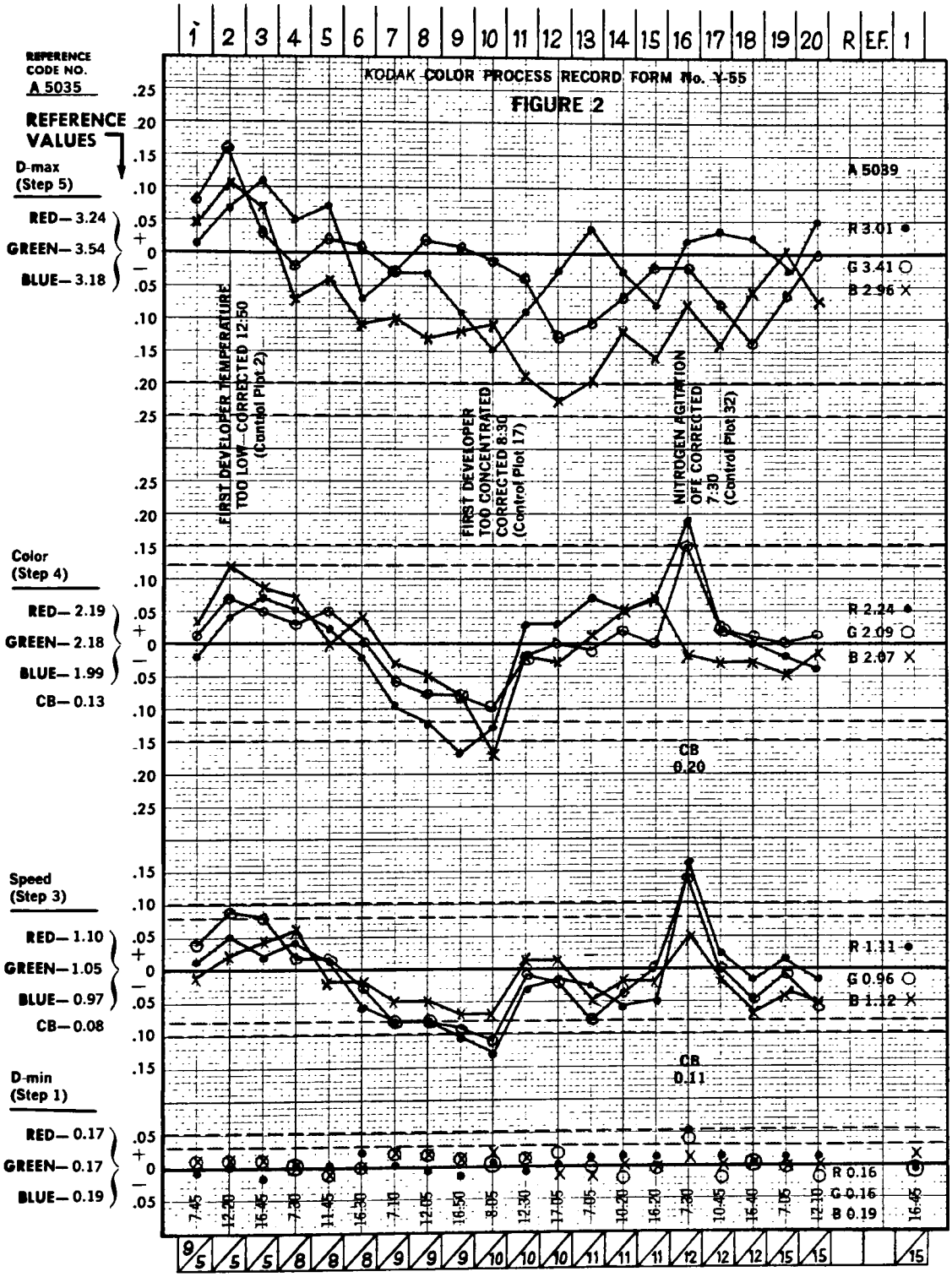
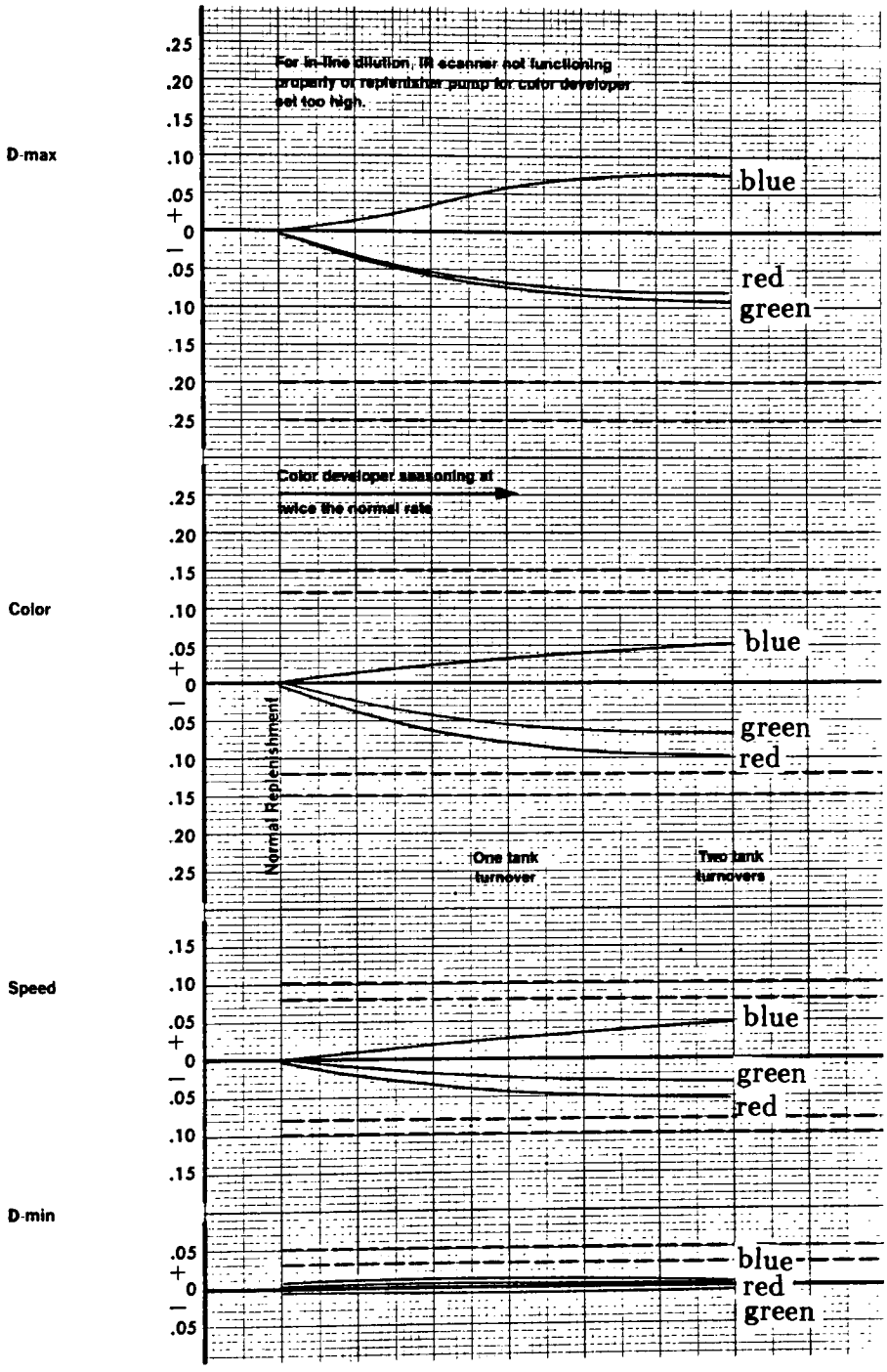


Figure 2.4: Densitometric effect of color developer overreplenishment



who are involved in photofinishing process diagnosis. Taking this idea one step further, it may be even more valuable to have tools available for photofinishing process experts to build diagnostic systems without any direct involvement of a knowledge engineer or any knowledge of the underlying expert system technology.

2.3 Diagnostic Expert Systems

Expert systems are special purpose computer programs which are "expert" in some narrow problem domain. Although loosely speaking, almost any problem solving program can fit into such a definition (for example, a program that is expert in doing arithmetic calculation), it is generally recognized that an expert system usually exhibits the following characteristics:

1. The problem solving strategies used are heuristic rather than algorithmic in nature.

2. The approach used generally mimics that of a human expert.

3. The knowledge is more explicitly represented in the program than by using conventional languages.

4. Explanation facilities are provided to allow users to judge the validity of the advice given.

Many expert systems have been built in the past ten years. Harmon[Harm 86] listed over one hundred expert systems actively in use today. Among them, about twenty percent are diagnostic systems.

The various domains of these diagnostic systems include human medical problems such as MYCIN[Shor 76] for diagnosing meningitis infections, agricultural problems such as PLANT[Mich 82] for soybean diseases, computer problems such as DART[Gene 83] for

computer hardware, chemistry/geology problems such as MUD[Kahn 86] for drilling fluid problems, and many others.

Among these diagnostic expert systems, the MUD System is the one closest in nature to the intended photofinishing process diagnostic system. The similarities between the two systems will become clear as we briefly examine the MUD System in the following section and the photofinishing processes diagnosis criteria in chapter three.

2.4 An Example Diagnostic Expert System — MUD

The MUD System (or MUD) is a drilling fluid diagnostic and treatment consultant system developed at Carnegie-Mellon University in cooperation with NL Baroid[Kahn 86]. MUD became a product in early 1985. It was built using OPS5 and has about 1600 rules.

A mud problem is defined as a departure from the expected measures of one or more of about twenty mud properties typically monitored by drilling fluids engineers. These properties include density, solids content, rheology, filtrate characteristics, etc. A diagnosis of a mud problem includes finding the causes for deviant test results. Examples of possible causes include contaminants, high temperatures and high pressures.

For any particular type of drilling fluid, there are roughly twenty or so significant potential causes of deviant mud properties. The number of evidential considerations relevant to the diagnosis of a given problem is usually between four and six. MUD may arrive at more than one hypothesis about possible causes for any set of test results, and hypotheses are ranked by confidence. MUD is able to explain its level of confidence and suggests treatments that may restore the mud properties.

In summary, it is apparent that MUD uses the evidential approach to diagnosis. That is, it by and large follows the following procedures:

1. Generate a set of plausible hypotheses.

2. Order the hypotheses for investigation.

3. For each hypothesis, determine the relevant evidential considerations.

4. Evaluate each hypothesis on the basis of the available evidence.

5. Accept or reject each hypothesis.

This evidential approach has proved to be successful for the implementation of MUD, and OPS5, a forward chaining rule-based expert system tool, was found to be a suitable tool for building such systems. The characteristics of OPS5 are described in Section 2.6.

2.5 Expert System Tools

There are many expert system building tools in use today to build expert systems. These tools range from special purpose ones, which are designed mainly for some particular applications, to general purpose ones, which contain a multitude of knowledge representation structures. Many of the tools were developed in universities and research institutes as research tools and have remained as such while some became commercially available or were developed solely for commercial purposes.

Waterman[Wate 86] listed about one hundred different expert system tools and gave a one-paragraph description for each one of the tools listed. He took a more liberal definition of expert system tools to include some procedural languages such as C, PROLOG and different versions of LISP. Many commercially unavailable research tools were also covered in his book.

Gilmore et al.[Gilm 85,Gilm 86] surveyed expert system tools and suggested some guidelines for evaluating these tools. Some of the guidelines used include the knowledge representation features available, the host machines that the tools can run on, the underlying language that the tools were developed with, user interface, etc.

Harmon[Harm 86] listed most of the commercially available expert system tools in his report. He tabulated the price, the approximate number the vendor claimed to have sold as of January, 1986, the number of fielded systems that have been built using that particular tool, and the hardware on which the tool runs. He also classified all the tools into eight categories, which may be grouped into four general categories. These four general categories are rule-based tools, inductive tools, domain specific tools and hybrid tools. Examples of rule-based tools and domain specific tools are discussed here because a rule-based tool is used for development in this project and the objective of this project is to design and implement a domain specific tool for building photofinishing process diagnostic systems.

2.6 An Example Rule-Based Tool — OPS5

Rule-based tools use IF-THEN rules as the basic knowledge representation technique. Two types of control strategies are available, namely, forward chaining and backward chaining. Forward chaining strategy starts from the data and works to the conclusion whereas backward chaining strategy starts from a stated goal and finds the plausibility of the goal by examining the conditions of the goal.

OPS5[Brow 85] is an example of a forward chaining rule-based expert system tool. It was developed at Carnegie-Mellon University as part of the OPS family of languages for expert systems and cognitive psychology. It is characterized by having a global data base and condition-action rules programmed in the form of productions which operate on the global data base.

The basic (and only) data structure in the global data base is called class which is equivalent to the idea of record in procedural languages. Each class has a class name and consists of a number of fields called attributes. An attribute may be either scalar which contains one atom or vector which contains a list of atoms. There are three types of atoms: symbols,

integers, and floating-point numbers. One restriction about class definition is that there may be at most one vector attribute per class.

Instances of classes are called elements. Elements can be created, modified or destroyed through actions which are the basic components of the right hand side of production rules.

Production rules in OPS5 employ forward chaining strategy. The left hand side of a production rule describes the conditions under which the rule may be activated. The conditions here refer to the current state of the data base. The right hand side of a production rule describes the actions to be taken when the rule is activated. In general, the actions will affect the current state of the data base and thus which rule to be activated next.

The rules can be regarded as independent parallel processes constantly monitoring the current state of the data base and waiting for the conditions necessary for them to become activated. Only one rule may be activated at any given time. When more than one rule have their conditions met, a conflict is said to occur. OPS5 resolves the conflicts by preferring the rule with more stringent conditions and/or conditions that match elements which are most recently created or modified. (Therefore, each element carries with it a time-stamp indicating the time it was created or last modified.) After the preferred rule is fired, the state of the data base may be changed and all rules have to re-check their conditions again for firing. This is known as the recognize-act cycles in OPS5.

OPS5 allows for interface with external routines written in other languages such as FORTRAN and PASCAL. It provides a very rudimentary file support facility allowing a program to save the current state of the data base and to retrieve elements from a file to replace or add to the current data base.

The support environment includes debugging facilities which allow the user to examine the current state of the data base, the time-stamps of the elements, and the sequence of the rule activation.

Other useful expert system support features such as certainty factor propagation and explanation facility are missing in OPS5. This means that it will probably take a longer time to build expert systems using OPS5 when such features are needed. On the other hand, this allows flexibility in designing one's own certainty factor propagation strategy and explanation facility.

2.7 An Example Domain Specific Expert System Tool — Micro IN-ATE

Domain specific tools are designed specifically to develop expert systems for a particular domain. Therefore these tools are most effective for the intended domain but may not be suitable for others.

IN-ATE is an example of such tool. There are two versions of IN-ATE: LISP IN-ATE, and Micro IN-ATE[ARC 85]. Both versions of IN-ATE were developed by Automated Reasoning Corporation. Micro IN-ATE is a scaled-down C version of the LISP IN-ATE to be run on the Apple Macintosh computer. It is an expert system environment specifically designed for fault diagnosis.

The knowledge engineer first inputs a schematic diagram of the device to be repaired (the so called "Unit Under Test" or "UUT") by drawing it on the computer screen with a mouse. This schematic should include the hierarchical structure or sub-structure of the UUT, the connections between structures, and the accessible testpoints.

Frames are provided for defining stereotypical UUT structures, connections, and testpoints. This is particularly useful for defining standard default values for failure rates, component replacement costs, and test costs.

Rules can be entered using the built-in interactive rule editor. This editor checks the consistency and correctness of the rules as they are created. Micro IN-ATE also can

automatically generate missing rules for the UUT. This is done first by deducing possible faults for symptoms and/or test results from the hierarchical structure, and then inducing probabilities for these possible faults using a statistical reliability database.

Micro IN-ATE learns as it is used. It remembers what has failed and automatically re-adjusts the reliability database and the rules. It can interactively guide a novice technician through each step of an actual troubleshooting session.

It also supports the IEEE-488 hardware bus to automate equipment testing and interfaces with digitized photograph or drawing tools in any standard PICT format file such as digitizing scanner, television signal digitizer, and MacPaint or MacDraw output files.

2.8 Summary

We have briefly described the photofinishing process control problems, expert systems for solving similar problems and domain specific expert system building tools. Since expert system technology has been so successfully employed in many applications including diagnosis of different problems in various fields, it would seem appropriate to apply this technology to the diagnosis of the control problem in a photofinishing process.

Chapter 3

Diagnosis Criteria for Photofinishing Process

3.1 Introduction

For the control of a photofinishing process, control film strips which are pre-exposed under controlled conditions are processed with customers' work. After a control strip is processed, a densitometer is used to read the red, green and blue densities of various exposure levels to determine the quality status of the process.

The densitometric readings of the control strip are then compared against the reference readings to measure the deviations of each color at each density step. When one or more of the deviations are beyond the action limits defined for that process, corrective actions will be taken to bring the process back under control.

To identify the underlying process control problem, process diagnosis experts generally analyze the control strip deviations available to describe the symptoms of the process. Customer's work is examined to verify that the symptoms suggested by the control strip are indeed present in the photofinishing results.

Process diagnosis experts also have accumulated knowledge about the possible causes for each symptom and the likelihood of each cause, and they use that knowledge to diagnose the process control problem.

3.2 Process E-6 Diagnostic System

Before the domain specific expert system building tool for photofinishing process was attempted, an expert system for diagnosing Process E-6 control problems was first built to understand the criteria and the underlying knowledge structures for photofinishing process diagnosis[Choi 86].

Process E-6 is a process for developing Kodak Ektachrome films. This prototype was chosen for two reasons. Firstly, it is a reversal process (that is, positive rather than negative images are produced), which is considered to be a rather complex process in comparison to Process C-41 and Process EP-2. However, it is not quite as involved as Process K-14 for processing Kodachrome films, so that there is a more limited set of possibilities and details.

Secondly, since Process E-6 has been out in the field for many years, it is relatively well understood and documented[EK 81] and there is a readily available pool of expertise in troubleshooting the process.

Figure 3.1 shows the logical structure of the Process E-6 diagnostic system. It uses an evidential approach to diagnosis. It starts from the control strip deviation readings and ends in suggesting the possible causes and their certainty factors through the following six phases:

1. Symptoms analysis.

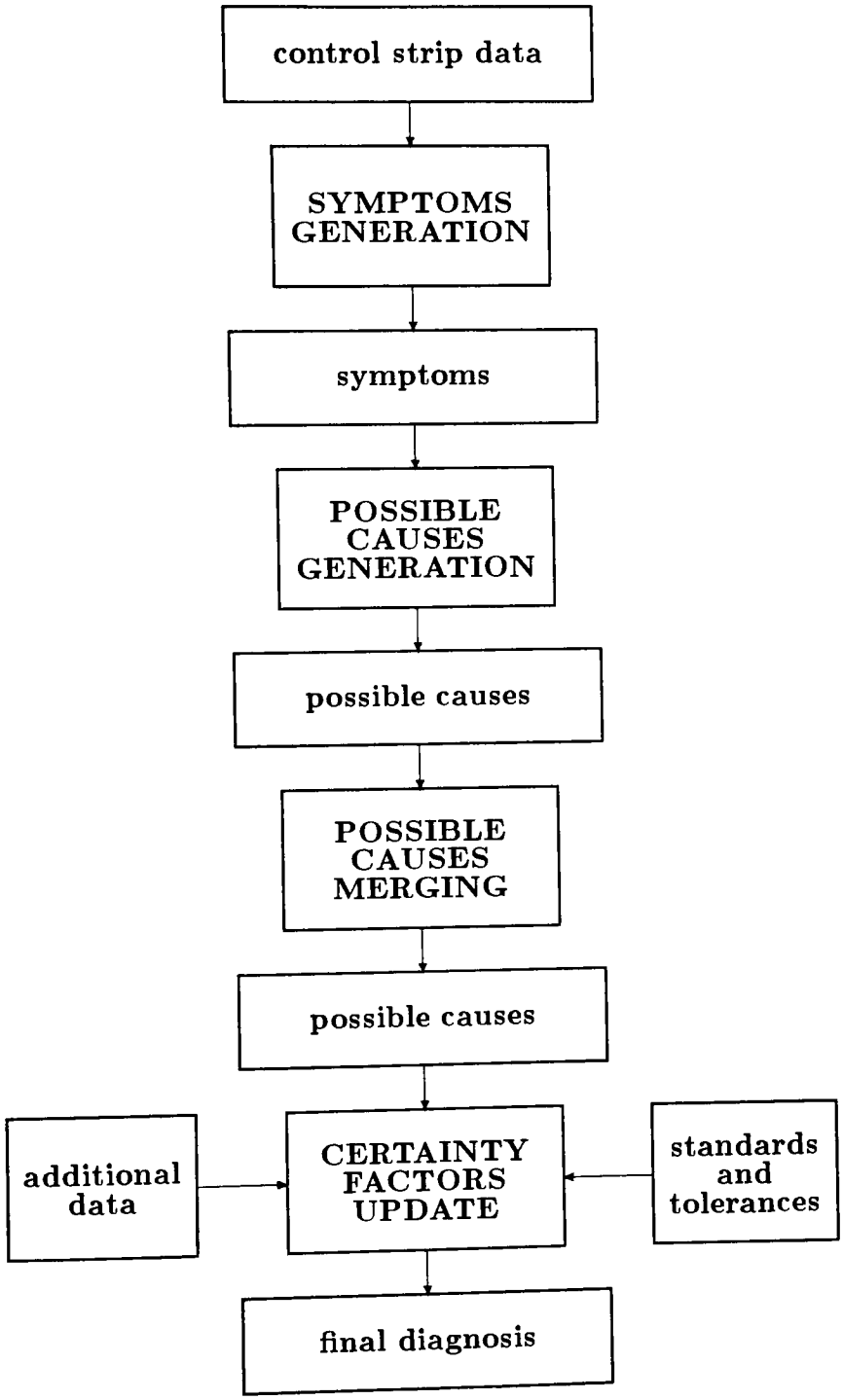
2. Possible causes generation.

3. Possible causes merging.

4. Additional data gathering.

5. Certainty factors update.

Figure 3.1: Inferencing structure of the prototype Process E-6 diagnostic system



6. Presentation of final diagnosis.

We will see that the first three phases constitute the hypotheses generation and ordering (steps one and two of the evidential approach outlined in Section 2.4). The last three phases correspond to the relevant evidential considerations determination and hypotheses evaluation and acceptance (steps three, four and five of the evidential approach).

3.2.1 Symptoms Analysis

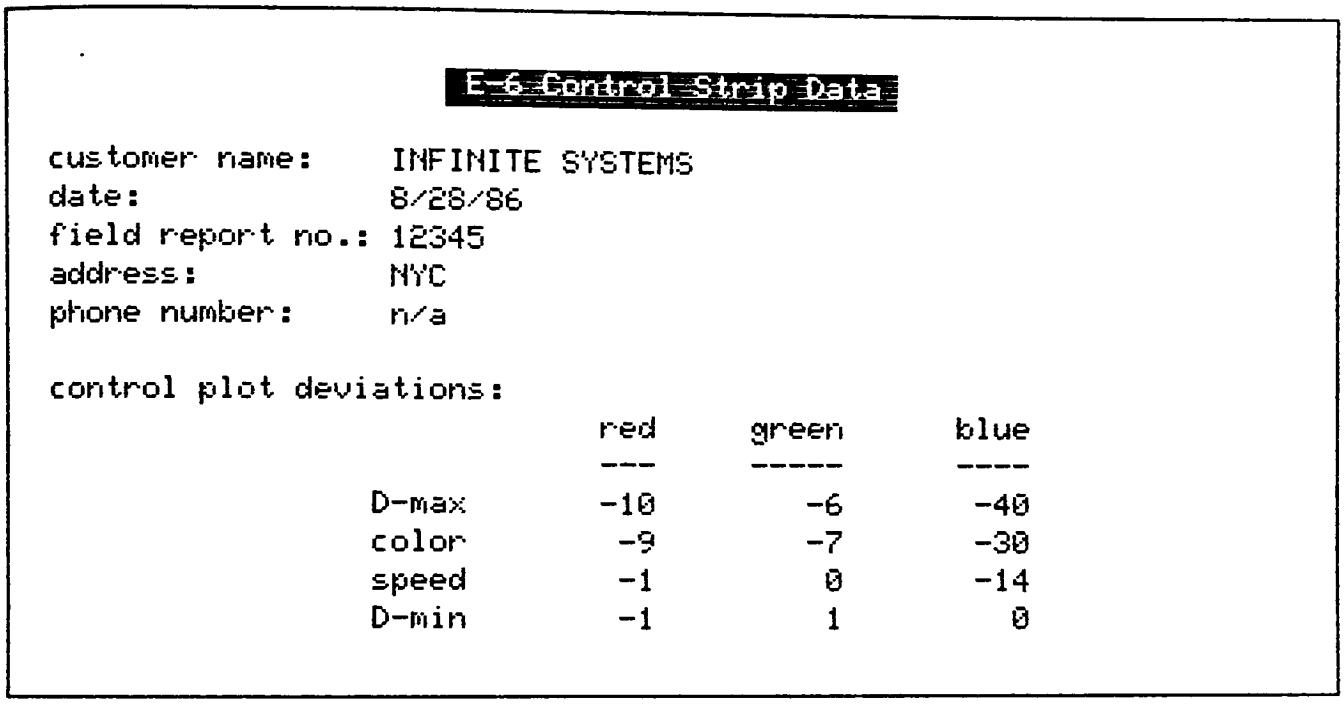
The control strip for Process E-6 contains twelve densitometric readings, which are the four exposure levels for each of the three colors red, green and blue. The four steps are generally known as the maximum density (D-max) step, the high density (color) step, the low density (speed) step and the minimum density (D-min) step.

These densitometric readings are then compared against the reference readings to find the deviation of each reading. Figure 3.2 is an example of the densitometric deviations. Each deviation reading is actually a two-digit decimal number less than one. However, the common practise for describing a deviation reading is to read it as a two-digit number without the decimal point. This practise is adopted in this system.

The diagnostic program also defines and calculates some derived parameters, which are functions of the control strip parameters. The purpose for having these derived parameters is to provide a better description of the status of the process and a convenient way to describe the symptoms. Figure 3.3 is a list of the possible symptoms and their conditions. For example, if the average D-max is less than zero, the blue D-max is less than the average D-max, and both the red D-max and the green D-max are greater than the average D-max, then we conclude that one of the symptom of this process is low blue D-max.

In order to describe the severity of the symptom, a certainty factor is assigned to each symptom. The range of the certainty factor is between zero and one, inclusively. It is a

Figure 3.2: Control strip data screen



Enter data ... Press one of the following keys
F6 Additional data F7 Diagnose F8 Save record

Figure 3.3: Symptoms and their conditions for Process E-6

	D-max				Color				Speed				D-min			
	red	green	blue	ave	red	green	blue	ave	red	green	blue	ave	red	green	blue	ave
	Rx	Gx	Bx	Ax	Rc	Gc	Bc	Ac	Rs	Gs	Bs	As	Rn	Gn	Bn	An
high D-max				>0												
low green red D-max	<Ax	<Ax	>Ax	<0												
low blue D-max	>Ax	>Ax	<Ax	<0												
low red D-max	<Ax	>Ax	>Ax	<0												
low blue green D-max	>Ax	<Ax	<Ax	<0												
low green D-max	>Ax	<Ax	<Ax	<0												
blue color balance					>Ac	>Ac	<Ac									
yellow color balance					<Ac	<Ac	>Ac									
magenta color balance					<Ac	>Ac	<Ac		<As	>As	<As					
green color balance					>Ac	<Ac	>Ac		>As	<As	>As					
yellow cyan color balance					<Ac	<Ac	>Ac		<As	>As	<As					
red color balance					<Ac	>Ac	>Ac		<As	>As	>As					
cyan color balance					>Ac	<Ac	<Ac		>As	<As	<As					
slow process								>0								
slow process								<0								
yellow neutral high D-min													<=An	<=An	>=An	>0
high contrast												<Ac				
low contrast												>Ac				

function of the underlying parameter's deviation from the aim. The greater the deviation, the higher the certainty.

3.2.2 Possible Causes Generation

After all the possible symptoms and their corresponding certainty factors are generated, the Process E-6 diagnostic system generates all the possible causes to these symptoms. Each possible cause is associated with a certainty factor. The certainty factor of a possible cause depends on the certainty factor of the deriving symptom and an attenuation factor. The values of the attenuation factors are supplied by Process E-6 troubleshooting experts.

Figure 3.4 is a partial list of the possible symptoms, the underlying possible causes and the corresponding attenuation factors. For example, in the course of diagnosing a control problem, the system deduces from the densitometric deviations that there is low green red D-max symptom in the process with a certainty factor of 60 percent. Looking up the attenuation factors for low green red D-max, there are three entries in this partial list. The possible causes that are generated are color developer contaminated with first developer with a certainty factor of 3 percent (60 percent times 5 percent), color developer pH too high with a certainty factor of 24 percent, and color developer has too much part A with a certainty factor of 30 percent.

3.2.3 Possible Causes Merging

When all the possible causes based on all the symptoms are generated, there might be possible causes with the same cause description generated from two or more different symptoms. This implies that this possible cause should be more likely. When such is the case, the possible causes merging phase combines the possible causes with the same cause description together and calculates a new certainty factor for the possible cause using the following certainty propagation formula:

Figure 3.4: The attenuation factors relating the symptoms and the possible causes

	SYMPTOM / DIAGNOSIS	1 BLEACH UNDERAERATED	2 C.D ADDED TO 1ST. D.	3 C.D. CONTAM. WITH 1ST D.	4 C.D. CONTAM. WITH FIXER	5 C.D. CONTAMINATION	6 C.D. TOO MUCH PART B.	7 C.D. OVERCONCENTRATED	8 C.D. OVER-REPLENISHED	9 C.D. OXIDISED	10 C.D. PH TOO HIGH	11 C.D. PH TOO LOW	12 C.D. REPL. TOO MUCH PART A	13 C.D. REPL. TOO MUCH PART B	14 C.D. START. ADD. TO 1ST. D.	15 C.D. STARTER TOO MUCH PART A.	16 C.D. TEMP. TOO HIGH	17 C.D. TEMP. TOO LOW	18 C.D. DILUTE
1	HIGH O-MAX																		60
2	LOW GREEN-RED O-MAX			5							40		50						
3	LOW BLUE O-MAX											60							
4	LOW RED O-MAX	80								60									
5	LOW BLUE-GREEN O-MAX																		
6	LOW GREEN O-MAX							40											
7	BLUE COLOUR BALANCE									1		70		10			1		
8	YELLOW COLOUR BALANCE		5			1			15		60					15		1	
9	MAGENTA COLOUR BALANCE																		80
10	GREEN COLOUR BALANCE							50											
11	YELLOW-CYAN COLOUR BALANCE			50	50														
12	RED COLOUR BALANCE	80																	
13	CYAN COLOUR BALANCE											50							
14	SLOW PROCESS						1			1					5				
15	FAST PROCESS							5	5										
16	YELLOW-NEUTRAL HIGH O-MIN																		
17	HIGH CONTRAST																		30
18	LOW CONTRAST							30											

$$CF = CF1 + CF2 - CF1 * CF2.$$

The properties of this formula are:

1. The new CF ranges between 0 and 1, inclusively.
2. The new CF is a mono-increasing function of $CF2$ given a fixed $CF1$, and vice versa.
3. The new CF is greater than both $CF1$ and $CF2$.
4. When there are more than two possible causes with the same cause description to be merged, the ordering of the merge is irrelevant.

Figure 3.5 shows an example of possible causes merging. In this example, there is low green red D-max symptom with certainty of 60 percent and yellow color balance with certainty of 50 percent. One of the possible causes derived from low green red D-max symptom is color developer pH too high with certainty of 24 percent. The same possible cause is derived from yellow color balance symptom with certainty of 30 percent. In merging the two possible causes together, we conclude that color developer pH too high has a new certainty factor of 47.2 percent ($0.24 + 0.30 - 0.24 * 0.30$).

3.2.4 Additional Data Gathering

When the generated possible causes can be confirmed or disconfirmed with additional data, the system will prompt the user to provide them. Upon the request of the system to provide additional data, the user can respond with any of the following:

1. Measure and enter the additional data.
2. Question why the additional data are relevant to the diagnosis.
3. Enter "n/a" to indicate that the data are not available.

Figure 3.5: An example of possible causes merging

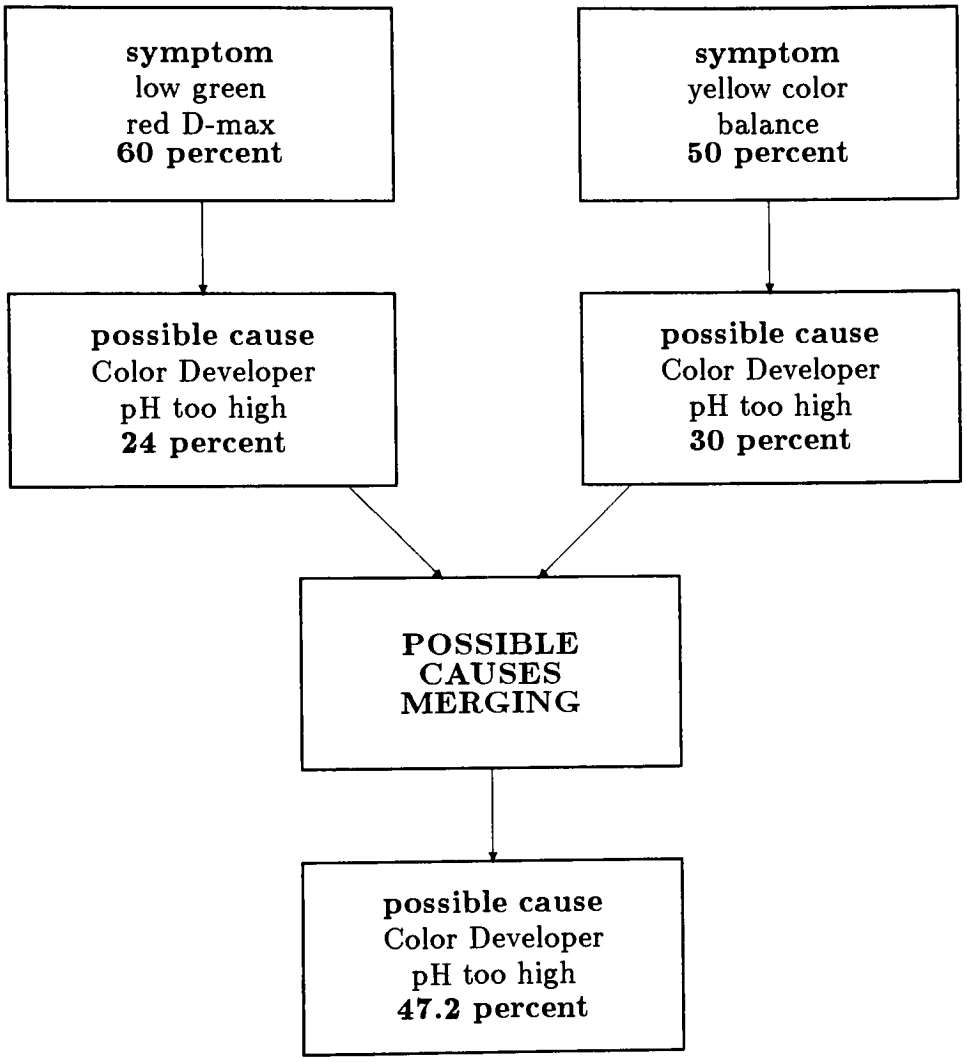
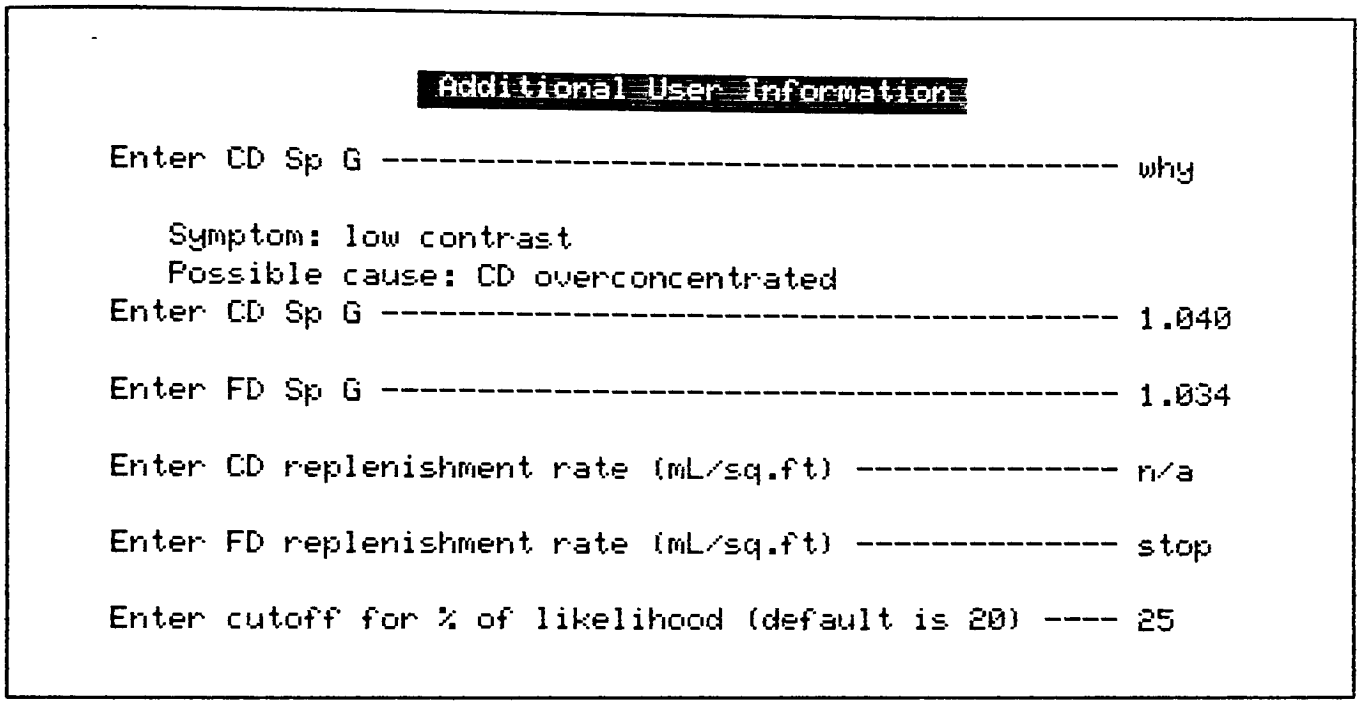


Figure 3.6: Additional data gathering



4. Stop the additional data gathering session.

Figure 3.6 is an example of a screen display during an additional data gathering session. Based on the deviation readings in Figure 3.2, the system prompted the user to enter color developer specific gravity. When the user questioned "why" this datum is needed, the system responded with the underlying symptom and the possible cause for that symptom. The user then entered the measurements of color developer specific gravity as 1.040.

Subsequently, the user entered 1.034 for first developer specific gravity, "n/a" for color developer replenishment rate, and stopped the additional data gathering when first developer replenishment rate was requested.

The user can also re-examine the values of the additional data entered and modify them if necessary. The additional data that are relevant to the diagnosis are flagged so that the user can take note and enter them later on.

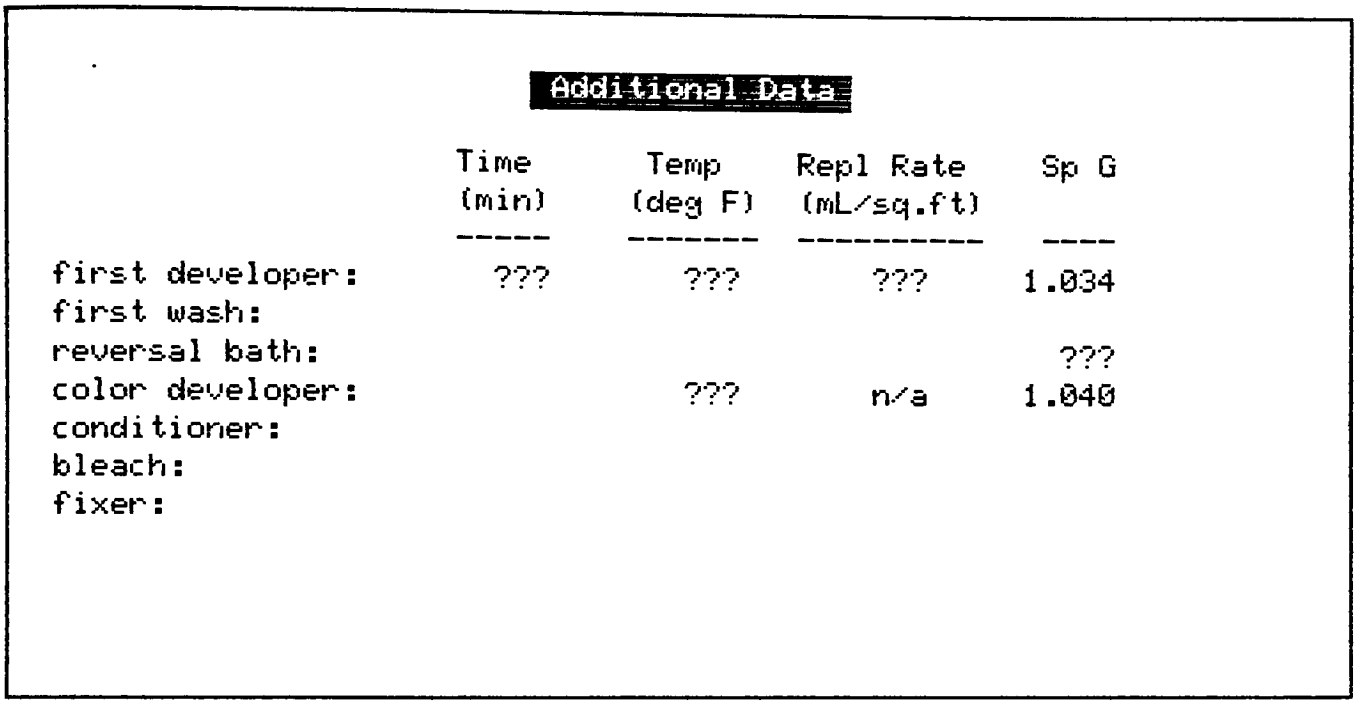
An example of the additional data screen display is shown in Figure 3.7. In this example, specific gravities of the first developer and color developer are 1.034 and 1.040, respectively, color developer replenishment rate is "n/a" and other data which are flagged as "???" are relevant but not yet known.

3.2.5 Certainty Factors Update

When relevant additional data are available, the system will derive modification factors to adjust the certainty factors of the possible causes generated. The range of the modification factors is between -1 for strongly disconfirming evidence, and 1 for strongly confirming evidence, inclusively.

Given a possible cause with a certain relevant additional datum, the modification factor MF is a function of the value of the additional datum V, the standard value of that additional datum S, the tolerance of that additional datum T, and the direction of the

Figure 3.7: Additional data screen



Modify data if needed ... Press one of the following keys
F5 Control strip data F7 Diagnose F8 Save record

confirming evidence D. The modification factor is derived using the following algorithm:

1. If D is high and $V \geq S + 2 * T$, then $MF = 1$.
2. If D is high and $S < V < S + 2 * T$, then $MF = (V - S - T)/T$.
3. If D is high and $V \leq S$, then $MF = -1$.
4. If D is low and $V \leq S - 2 * T$, then $MF = 1$.
5. If D is low and $S > V > S - 2 * T$, then $MF = (S - T - V)/T$.
6. If D is low and $V \geq S$, then $MF = -1$.

The properties of the above algorithm are:

1. MF is between -1 and 1, inclusively.
2. MF is a continuous monotonic function of V.
3. If V is opposite to the confirming evidence direction, $MF = -1$.
4. If V is within the tolerance, MF is negative (disconfirming).
5. If V is beyond the tolerance in the confirming evidence direction, MF is positive (confirming).

Once the modification factors are derived, the new certainty factors can be calculated using the following algorithm:

1. If $0 \leq MF \leq 1$, then $CF = CF + MF - CF * MF$.
2. If $-1 < MF < 0$, then $CF = (CF + MF)/(1 + MF)$.
3. If $MF = -1$, then $CF = -1$.

The properties of the above algorithm are:

1. CF is between -1 and 1, inclusively.

2. The new CF is a continuous monotonic function of MF and the old CF.

3. CF remains unchanged when MF is zero.

Figure 3.8 shows an example of certainty factor update. In this example, the certainty factor of the possible cause color developer replenishment too high is 60 percent. The standard replenishment rate is 200 milliliters per square foot. The tolerance is 20 milliliters per square foot. The direction of confirming evidence is high. Given that the value of the color developer replenishment rate is 230 milliliters per square foot, the modification factor is 50 percent $((230 - 200 - 20)/20)$. The new certainty factor is thus 80 percent $(0.60 + 0.50 - 0.60 * 0.50)$.

Possible causes for which there is no relevant additional data or whose relevant additional data are not available will have their certainty factors unmodified.

3.2.6 Presentation of Diagnosis

After all the certainty factors of the possible causes have been updated, the system prompts the user to enter the cutoff percentage of certainty factor. Possible causes with certainty factors greater than the user's input cutoff are displayed to the user in descending order of certainty factors.

Figure 3.9 shows a typical listing of possible causes based on the deviation readings in Figure 3.2 and the additional values in Figure 3.7. In this example, the most possible cause is color developer overconcentrated. The next possible causes are color developer pH low followed by color developer underreplenished.

Figure 3.8: Certainty factor update

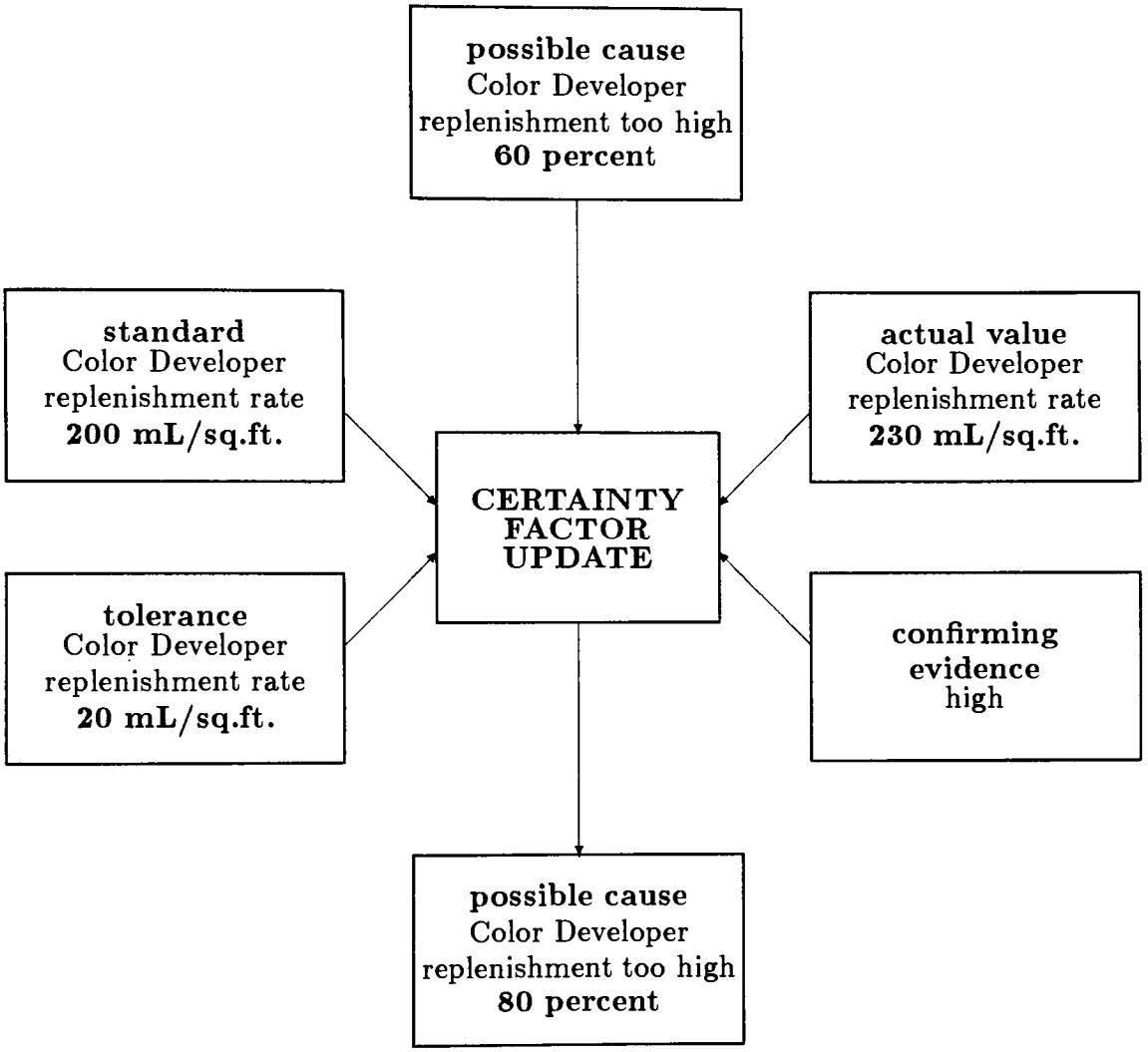
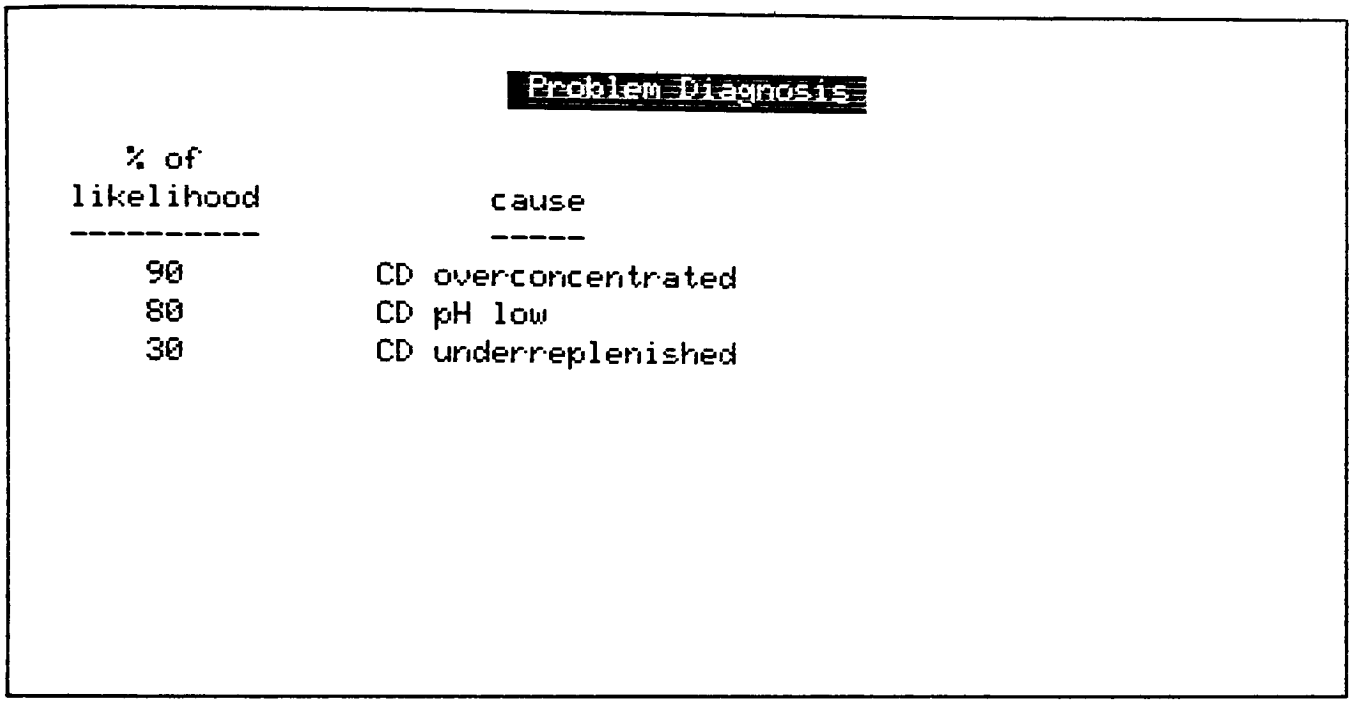


Figure 3.9: A typical listing of possible causes



3.2.7 Summary

The prototype Process E-6 diagnostic system was built in a VAX/VMS environment using OPS5 as the inference engine supplemented by FORTRAN routines for number crunching and user interface. It took about two months of development time.

To facilitate the user interface, the VAX screen management system was used to provide a menu-driven input/output environment. It turned out that over fifty percent of the implementation time was spent on building a good user interface.

A trial system was set up for use and evaluation. In this trial system, a micro-VAX was used with four terminals connected to it. The performance was excellent. There is no noticeable delay in response time to any user requests.

Initial comments indicate that the system has captured most or all of the possible causes without missing any major ones. However, the general response from the personnel who assist processing laboratories to solve their process control problems, is that the prototype is only somewhat helpful. The approach used is very appropriate but more knowledge should be incorporated into the prototype to make it useful at a more sophisticated level.

It was estimated that at least another six months of the knowledge engineer's time and three months of a Process E-6 expert's time would be required to upgrade this prototype to a level that would provide the correct responses ninety percent of the time.

Suggestions were also made to apply the same technology to assist the diagnosis of other photofinishing processes such as Process C-41 and Process K-14. Thus, having an expert system tool for building diagnostic systems for different photofinishing processes would be an ideal solution to future development and maintainence of such diagnostic systems.

Chapter 4

Photofinishing Process Diagnostics Expert System Tool

4.1 Introduction

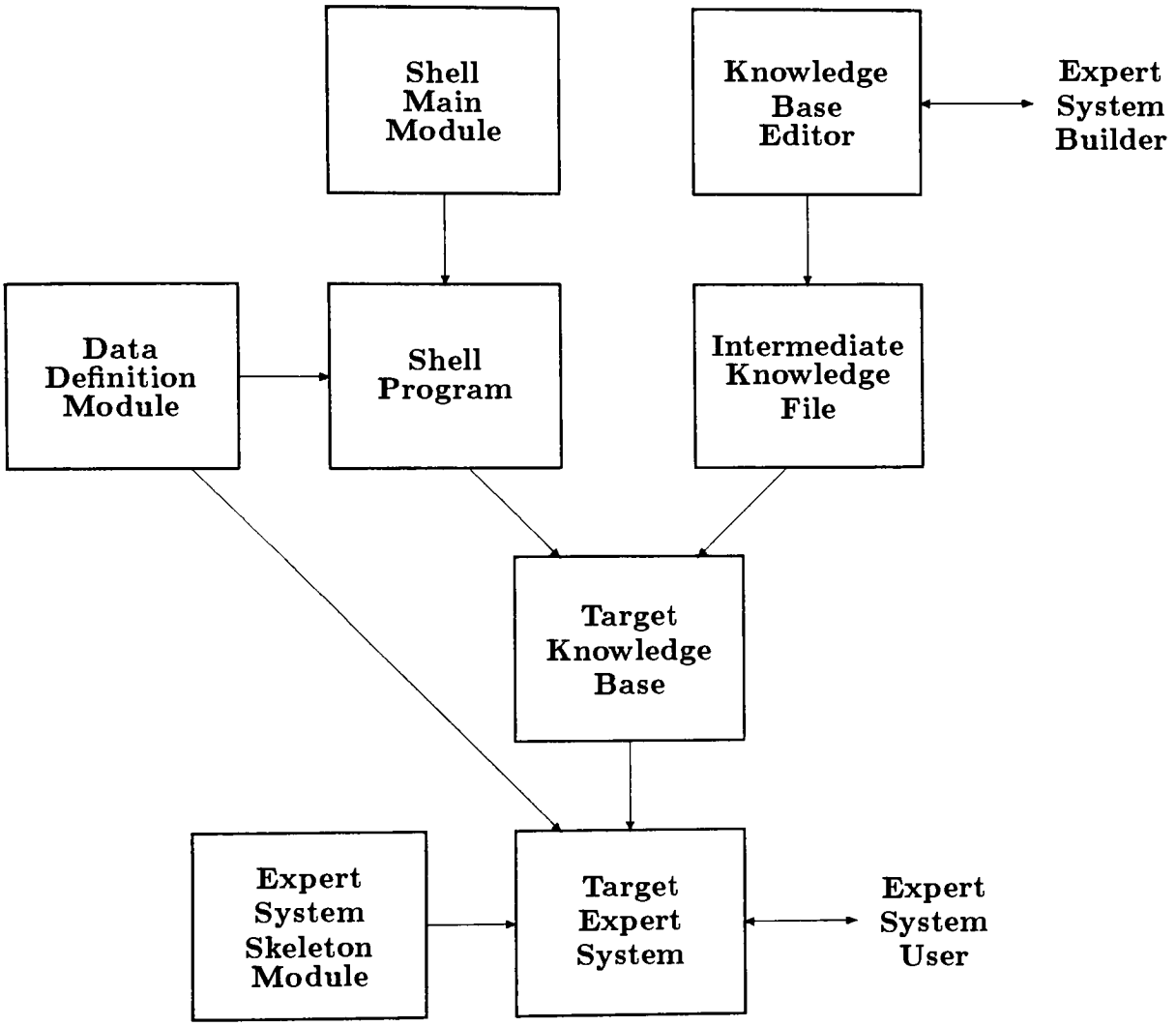
The experience and knowledge gained from the prototype Process E-6 diagnostic system was applied to design the photofinishing process diagnostic expert system tool.

This tool consists of four main components, namely, the knowledge base editor, the data definition module, the shell main module and the expert system skeleton module. Figure 4.1 shows the structure of the expert system tool and how the components relate to each other.

In building a target expert system, an expert system builder first creates an intermediate knowledge file. This intermediate knowledge file is in some peculiar syntactic format reflecting the underlying implementation language used. Because of this, assistance to the target expert system builder is provided through the use of a knowledge base editor. This knowledge base editor interfaces with the user in a "friendly" way and produces the desired intermediate knowledge file.

The intermediate knowledge file serves as an input to the shell program to create the target knowledge base. The shell program can be divided into two modules, namely, the data definition module and the shell main module. The data definition module contains

Figure 4.1: Software architecture of the expert system tool



data structures that are common to the expert system building stage as well as the target system consultation stage. The shell main module contains routines which use the data structures defined in the data definition module to convert the intermediate knowledge file to a target knowledge base.

The target knowledge base and the data definition module are then combined with the expert system skeleton module to form the target expert system.

4.2 Knowledge Base Editor

The purpose of the knowledge base editor is to provide a user interface for an expert system builder to create and modify the target knowledge base of a target expert system. It shields the expert system builder from the internal design of the knowledge structure and the language(s) used to implement the target knowledge base. The minimal functionalities of the knowledge base editor are as follows:

1. Add new instances of the pre-defined frames.
2. Delete instances of the pre-defined frames.
3. Modify instances of the pre-defined frames.
4. List instances of the pre-defined frames.
5. Provide help facilities to guide expert system builders when necessary.
6. Update the knowledge base at the end of an editing session.
7. Terminate an editing session without updating.

The output of the knowledge base editor is an intermediate knowledge file which is used as an input to the shell program to produce a target knowledge base of the target expert system.

Figure 4.2 shows how this editor is invoked and the commands that are available. It is basically a line editor providing a front end interface for the expert system builder.

4.3 Data Definition Module

4.3.1 Overview

The data definition module contains all the common data structure definitions for building the shell program as well as the target expert system program. It includes a number of data definitions to represent the following frames:

1. Control strip parameter frame.
2. Derived parameter frame.
3. Additional data frame.
4. Symptom description frame.
5. Cause-solution description frame.
6. Conflicting cause-pair frame.

4.3.2 Control Strip Parameter Frame

Since photofinishing process monitoring and diagnosis start out from the data on the control strip, we need control strip parameter frames to represent these data. A control strip parameter frame consists of the following slots:

1. The control strip parameter name slot.
2. The deviation value slot.

Figure 4.2: Help command in the knowledge base editor

```
run kbe

Enter Process Name (1-8) characters: e6

KBE> help

type a <frame_type> to add frame
type d <frame_type> <frame_no> to delete frame
type h for help
type l <frame_type> [<frame_no>] to list frame(s)
type m <frame_type> <frame_no> to modify frame
type q to quit (without update)
type x to exit (with update)

possible choices of <frame_type> are:
  csp --- control strip parameter
  ddp --- derived parameter
  add --- additional data
  csd --- cause & solution description
  ccp --- conflicting cause pair
  sym --- symptom description

KBE>
```

The control strip parameter name slot identifies the name of that control strip parameter. Its value is to be entered during the target expert system building stage.

The deviation value slot contains the densitometric deviation of that parameter from the aim or reference value. It is to be entered by an end-user of the target expert system during the consultation stage.

Figure 4.3 is a list of twelve control strip parameter frame instances using the knowledge base editor. Note that only the name slots are listed since the values of the deviation value slots are not yet known.

4.3.3 Derived Parameter Frame

It is often useful to examine some derived parameters, reflecting the overall behavior of a group of parameters, than to examine individual parameters. This can be achieved by defining derived parameter frames with the following slots in each frame:

1. The derived parameter name slot.
2. The deriving formula slot.
3. The parameter value slot.

The derived parameter name slot identifies the name of that derived parameter. Its value is to be entered during the target expert system building stage.

The deriving formula slot describes how the value is to be calculated. The format of this formula is like a typical arithmetic expression relating the control strip parameters or other derived parameters. This slot is to be entered during the target expert system building stage.

The parameter value slot contains the actual value of that derived parameter. It is to be generated by the target expert system during the consultation stage when all the deriving

Figure 4.3: List of control strip parameter frames

```
KBE> l csp

csp 1: red_D-max
csp 2: green_D-max
csp 3: blue_D-max
csp 4: red_Color
csp 5: green_Color
csp 6: blue_Color
csp 7: red_Speed
csp 8: green_Speed
csp 9: blue_Speed
csp 10: red_D-min
csp 11: green_D-min
csp 12: blue_D-min

KBE>
```

control strip parameters and/or derived data parameters are known.

Figure 4.4 is a list of four derived parameter frame instances using the knowledge base editor. Taking the first frame instance as an example, the name slot has a value of ave_D-max and the formula slot indicates that the value of ave_D-max can be derived by summing up the red_D-max, the green_D-max, and the blue_D-max and then dividing by three.

4.3.4 Additional Data Frame

Additional data, as they are used here in this paper, refer to the chemical data (such as specific gravity and the concentration of a certain chemical in a processing solution) and the physical data (such as the temperature and replenishment rate of a processing solution). From the view point of the operation of a photofinishing process, these are the basic parameters in controlling the process. However, from a process diagnostic point of view, these data may or may not be available, and often troubleshooting recommendations have to be made even in the absence of these data. An additional data frame includes the following slots:

1. The name of the additional datum.
2. The standard value of the additional datum.
3. The tolerance limit around the standard value.
4. The actual value of the additional datum.

The additional datum name slot identifies the name of that additional datum. Its value is to be entered during the target expert system building stage.

The standard value slot identifies the reference operating point of that additional datum. Photofinishing process designers usually make recommendations about these reference operating points. However, different processing laboratories may want to deviate from such

Figure 4.4: List of derived parameter frames

```
KBE> l ddp

ddp 1:
  name: ave_D-max
  formula: ( red_D-max + green_D-max + blue_D-max ) / 3
ddp 2:
  name: ave_Color
  formula: ( red_Color + green_Color + blue_Color ) / 3
ddp 3:
  name: ave_Speed
  formula: ( red_Speed + green_Speed + blue_Speed ) / 3
ddp 4:
  name: ave_D-min
  formula: ( red_D-min + green_D-min + blue_D-min ) / 3

KBE>
```

recommendations for their own reasons. Thus, an expert system builder can tailor make his own reference operating points by entering the appropriate values during the target expert system building stage.

The tolerance slot identifies the operating region of that additional datum. Again, photofinishing process designers make recommendations about these operating regions, while different processing laboratories may want to deviate from such recommendations.

The additional datum value slot contains the actual value of that parameter. It is to be entered by an end-user of the target expert system during the consultation stage at the prompt of the system.

Figure 4.5 is a list of eight additional data parameter frame instances using the knowledge base editor. Taking the first frame instance as an example, the name slot has a value of first developer time. The standard value slot states that six minutes is the standard for first developer time. The tolerance slot indicates that the actual first developer time should stay within one minute around the stardard time.

4.3.5 Symptom Description Frame

Different photofinishing processes may have different symptoms associated with each one of them. The symptom description frame is used to name the symptom, describe the conditions ascribed to the symptom, and give possible underlying causes of the symptom. A symptom description frame contains the following slots:

1. The name of the symptom.

2. A variable number of slots for the conditions of the symptom.

3. A variable number of slots for the possible underlying causes.

Figure 4.5: List of additional data frames

```
KBE> l add

add 1:
  name: First Developer time
  standard: 6.0
  tolerance: 1.0
add 2:
  name: First Developer temperature
  standard: 100.4
  tolerance: 2.5
add 3:
  name: First Developer replenishment rate
  standard: 200.0
  tolerance: 20.0
add 4:
  name: First Developer specific gravity
  standard: 1.059
  tolerance: 0.01
add 5:
  name: First Wash time
  standard: 2.0
  tolerance: 0.2
add 6:
  name: First Wash temperature
  standard: 97.5
  tolerance: 5.5
add 7:
  name: First Wash replenishment rate
  standard: 0.0
  tolerance: 0.0
add 8:
  name: First Wash specific gravity
  standard: 1.0
  tolerance: 0.01
```

The symptom name slot identifies the name of that symptom. Its value is to be entered during the target expert system building stage.

The condition slots are relational expressions relating the control strip parameters and/or the derived parameters to the symptom. These conditions are expressed as relational expressions with an implicit logical AND joining them together. They are to be entered during the target expert system building stage.

The possible underlying cause slots identify the possible causes of the symptom and their corresponding attenuation factors. Thus, each slot actually consists of two sub-slots. The first sub-slot contains the description of the possible cause and the second sub-slot contains the attenuation factor of that cause. Both sub-slots are to be entered during the target expert system building stage.

Figure 4.6 is a list of two symptom description frame instances using the knowledge base editor. Taking the first frame instance as an example, the symptom name slot has a value of color developer too dilute. There is only one condition slot which states that the condition for this symptom is the average D-max greater than zero. There are six possible causes slots. They are color developer too dilute, first developer underreplenished, color developer underreplenished, first developer too dilue, first developer and color developer reversed and first developer omitted with the correspondingly listed attenuation factors.

4.3.6 Cause-Solution Description Frame

The cause-solution description frame contains information relating a possible cause to the relevant additional data, the evidence, and the solution to the control problem for that particular cause. A cause-solution description frame includes the following slots:

1. The name of the possible cause.

2. The relevant additional datum concerning the possible cause.

Figure 4.6: List of symptom description frames

```
KBE> l sym

sym 1:
  symptom: high D-max
  condition 1: ave_D-max > 0
  possible cause 1: Color Developer too dilute
  attenuation factor: 60
  possible cause 2: First Developer underreplenished
  attenuation factor: 25
  possible cause 3: Color Developer underreplenished
  attenuation factor: 25
  possible cause 4: First Developer too dilute
  attenuation factor: 15
  possible cause 5: First Developer & Color Developer reversed
  attenuation factor: 01
  possible cause 6: First Developer omitted
  attenuation factor: 01
sym 2:
  symptom: low green-red D-max
  condition 1: ave_D-max < 0
  condition 2: red_D-max < ave_D-max
  condition 3: green_D-max < ave_D-max
  possible cause 1: Color Developer replenisher has too much part A
  attenuation factor: 50
  possible cause 2: Color Developer pH high
  attenuation factor: 40
  possible cause 3: Color Developer contaminated with First Developer
  attenuation factor: 05
  possible cause 4: Too little Color Developer starter
  attenuation factor: 05
```

3. The confirming evidence of the additional datum.

4. The solution prescribed to fix the problem.

The possible cause name slot identifies the name of that possible cause. Its definition is to be entered during the target expert system building stage.

The relevant additional datum slot provides a linkage between the cause-solution description frame and the additional data frame. When a possible cause is generated during a target expert system consultation session as a hypothesis, if the relevant additional datum slot is not empty, the system prompts the end-user to enter the value of that additional datum and checks the standard and tolerance slots of the additional data frames to update the certainty of that possible cause. This slot is to be entered during the target expert system building stage.

The evidence slot is an indicator on which direction (high or low) of the additional datum's deviation from the standard constitutes the confirming evidence. This slot is to be entered during the target expert system building stage.

The solution slot contains a description of how to fix the problem concerning that possible cause. This slot is to be entered also during the target expert system building stage.

Figure 4.4 is a list of seven cause-solution description frame instances using the knowledge base editor. Taking the sixth frame instance as an example, the possible cause name slot has a value of color developer overconcentrated. The relevant data slot indicates that the specific gravity of the color developer is a relevant datum to this possible cause. The confirming evidence slot shows that a high value of the specific gravity is the confirming evidence. The solution slot suggests that adding water to the color developer may solve the problem.

Figure 4.7: List of cause-solution description frames

```
KBE> l csd

csd 1:
  cause: Bleach underaerated
  relevant data: none
  evidence: none
  solution: Increase bleach aeration
csd 2:
  cause: Color Developer contaminated with First Developer
  relevant data: none
  evidence: none
  solution: Dump First Developer and start afresh
csd 3:
  cause: Color Developer contaminated with Fixer
  relevant data: none
  evidence: none
  solution: Dump First Developer and start afresh
csd 4:
  cause: Color Developer contamination
  relevant data: none
  evidence: none
  solution: Dump Color Developer and start afresh
csd 5:
  cause: Color Developer has too much part B
  relevant data: none
  evidence: none
  solution: Dump Color Developer and start afresh
csd 6:
  cause: Color Developer overconcentrated
  relevant data: Color Developer specific gravity
  evidence: high
  solution: Add water to Color developer
csd 7:
  cause: Color Developer overreplenished
  relevant data: Color Developer replenishment rate
  evidence: high
  solution: Decrease Color Developer replenishment rate to 200 mL/sq.ft
```

4.3.7 Conflicting Cause-pair Description Frame

It is possible that two different symptoms in a photofinishing process control problem point to two conflicting possible causes. Instead of allowing both conflicting causes to be presented to an end-user during consultation, the system identifies the conflicting possible causes and resolves the conflict by weighing the certainty factors associated with these causes. The possible cause with a lower certainty factor say $CF1$ will be eliminated while the one with higher certainty factor say $CF2$ will remain but with its certainty reduced to $(CF2 - CF1)/(1 - CF1)$.

This formula is compatible with the one described in the section on possible causes merging (Section 3.2.3). For example, if $CF = CF1 + CF2 - CF1 * CF2$, then $CF1 = (CF - CF2)/(1 - CF2)$ and $CF2 = (CF - CF1)/(1 - CF1)$.

A conflicting cause-pair description frame consists of the following slots:

1. The name of the first possible cause.

2. The name of the second possible cause.

Both slots together identify a conflicting possible cause-pair and they are to be entered during the target expert system building stage.

Figure 4.8 is a list of eleven conflicting cause-pair frame instances using the knowledge base editor. Taking the first frame instance as an example, color developer overconcentrated and color developer too dilute are two conflicting possible causes. If during the consultation stage, both possible causes are suggested with certainty factors 80 percent and 60 percent respectively, then color developer too dilute is eliminated while color developer overconcentrated will have a new certainty factor equals to 50 percent $((.80 - .60)/(1 - .60))$.

Figure 4.8: List of conflicting cause-pair frames

```
KBE> l ccp

ccp 1:
  cause1: Color Developer overconcentrated
  cause2: Color Developer too dilute
ccp 2:
  cause1: Color Developer overreplenished
  cause2: Color Developer underreplenished
ccp 3:
  cause1: Color Developer pH high
  cause2: Color Developer pH low
ccp 4:
  cause1: Color Developer replenisher has too much part A
  cause2: Color Developer replenisher has too much part B
ccp 5:
  cause1: Color Developer temperature high
  cause2: Color Developer temperature low
ccp 6:
  cause1: First Developer overconcentrated
  cause2: First Developer too dilute
ccp 7:
  cause1: First Developer overreplenished
  cause2: First Developer underreplenished
ccp 8:
  cause1: First Developer temperature high
  cause2: First Developer temperature low
ccp 9:
  cause1: First Developer time too long
  cause2: First Developer time too short
ccp 10:
  cause1: First Wash temperature high
  cause2: First Wash temperature low
ccp 11:
  cause1: Too little Color Developer starter
  cause2: Too much Color Developer starter
```

4.4 Shell Program

The purpose of the shell program is to take the intermediate knowledge file produced by the knowledge base editor as an input and produce the target knowledge base of the target expert system. The shell program consists of the following two modules:

1. The data definition module.

2. The shell main module.

The data definition module contains the definitions of various frames as described in Section 4.3.

The shell main module contains routines that take the intermediate knowledge file as input and generate the target knowledge base of a target expert system.

4.5 Target Expert System

A target expert system is the desired output of the expert system builder using this expert system building tool. It is the program that will be used by end-users for photofinishing process control problem consultation. The following are the requirements for the target expert system:

1. Prompt the end-users to provide control strip parameters. For those control strip parameter instances defined during the expert system building stage, the target expert system tries to fill up the unknown parameter value slots during the consultation stage.

2. Calculate the derived parameters. Based on the derived parameter frame instances defined during the expert system building stage, once the underlying deriving param-

eters are available, the target expert system will calculate the values of the derived parameters.

3. Generate the symptoms and the possible causes. With the control strip parameter values and the derived parameter values known, the target expert system will deduce all the symptoms and the possible causes.

4. Prompt the end-users to provide the relevant additional data. When there are possible causes that can be further confirmed or disconfirmed with additional data, the target expert system will request those data.

5. Generate and propagate certainty factors. As symptoms are generated, possible causes deduced and merged, and additional data gathered, the target expert system maintains the certainty factors and propagates them to the final presentation stage.

6. Provide explanation facilities. During consultation, the target expert system allows the user to question why certain additional data are needed or how the system arrives at its conclusions.

7. Present final diagnosis to the end-users. When the diagnosis is done, the target expert system presents the final diagnosis of the control problem to the end user.

The target expert system consists of the following three components:

1. The data definition module.

2. The target knowledge base.

3. The expert system skeleton module.

The data definition module contains the definitions of various frames as described in the Section 4.3.

The target knowledge base is the output of the shell program based on the input from an expert system builder.

The expert system skeleton module contains control structures that use the data definition module and the target knowledge base to diagnose the presented photofinishing process control problem and to provide explanations to user queries.

Chapter 5

Implementation

5.1 Introduction

After the design of the logical structure of the photofinishing process diagnostic expert system tool is in place, the next step is to implement the tool. There are three main questions for the implementation of this tool:

1. What system or environment should the tool be implemented on?
2. What language(s) should be used?
3. What should the user interface be like?

First, we address the question of what language should be used. The expert system building tools available today are first divided into three categories:

1. Generic AI languages such as LISP or PROLOG.
2. Rule-based expert system tools such as EMYCIN or OPS.
3. Expert system tools with multi-knowledge representation schemes such as Knowledge Engineering Environment (KEE) or Automated Reasoning Tool (ART).

The generic languages were first ruled out because they require much more time and effort of development before we can see any results. The multi-knowledge representation scheme tools were also eliminated mainly because of lack of availability.

Among the choices in the second category, OPS5 was chosen for four reasons. Firstly, it is available. Secondly, it is a tool that has been proven in building expert systems. Thirdly, the Process E-6 diagnostic system was built using OPS5 and many of its knowledge structures can be used to build this expert system tool with minimal effort. Finally, in building the Process E-6 diagnostic system, experience in the language was gained. Thus, the decision on the choice of language was not difficult.

The only exception to the above discussion is the language to be used to build the knowledge base editor. Since this piece of the system is only loosely linked with the other components of the tool through the intermediate knowledge file, we can use a different language to implement the editor. (In fact, OPS5 would certainly be a wrong tool to build such editor.) Among many possibilities, C was chosen to do the job.

Once the language was decided, the choice of system or environment became obvious — whatever OPS5 is running on at my desktop. So the VAX/VMS environment was chosen.

Finally, the last question is about the user interface. Experience tells us that we can spend a lot of time to dress up the user interface. Since the main thrust of this project was to try out the idea of building an expert system building tool for photofinishing process diagnostics, the user interface became a "necessary evil" of this project, and I tried to keep it minimal. Nevertheless, some efforts were made, such as building the knowledge base editor, to provide a user interface to build the target system, and providing help facilities to guide the end-users in using a target expert system.

5.2 Knowledge Base Editor

As described in Section 4.2, the purpose of the knowledge base editor is to provide a convenient interface for an expert system builder to create and modify the target knowledge base. The output of the knowledge base editor is an intermediate knowledge file. This program is written in C with six input files corresponding to the six pre-defined frames, and one output file — namely, the intermediate knowledge file.

5.3 Data Definition Module

All the pre-defined frames are represented as classes of working memory elements in OPS5. An example of a working memory element definition in OPS5 is as follows:

```
(LITERALIZE CONFLICTING_CAUSE_PAIR CAUSE1 CAUSE2)
```

In OPS5, literalize is a keyword for defining working memory elements; the second member of the list conflicting_cause_pair is the class name; and the remaining members of the list such as cause1 and cause2 are attributes of the working memory element.

5.4 Shell Main Module

5.4.1 Overview

The shell main module is written in OPS5. Its function is to read the intermediate knowledge file and create the target knowledge base. This is done in the following six phases:

1. Generate the control strip parameter frame instances.

2. Generate the derived parameter frame instances and the corresponding value deriving production rules.

3. Generate the additional data frame instances.

4. Generate the symptom description frame instances, production rules for determining the symptoms from the conditions and production rules for deducing the possible causes.

5. Generate the cause-solution description frame instances.

6. Generate the conflicting cause-pair description frame instances.

5.4.2 Frame Instances

The first, third, fifth and sixth phases are very similar and straight forward. A typical frame instance that is generated is as follows:

```
(CONFLICTING_CAUSE_PAIR ^CAUSE1 6 ^CAUSE2 18)
```

In OPS5, this is an instance of a working memory element with conflicting_cause_pair being the class name, cause1 and cause2 attribute types, and 6 and 18 the contents of the attributes of indices pointing to the cause descriptions.

5.4.3 Production Rules for Derived Parameters

In the second phase, besides generating the derived parameter frame instances, production rules for deriving the values of parameters are also generated. The following is a typical example of such a production rule:

```
(P G:1
     (CONTEXT ^STATE CAL_DP)
     { (CSP ^ID AVE_D-MAX ^VAL NIL) <CSP> }
     (CSP ^ID RED_D-MAX ^VAL { <P0> <=> 0 } )
     (CSP ^ID GREEN_D-MAX ^VAL { <P1> <=> 0 } )
     (CSP ^ID BLUE_D-MAX ^VAL { <P2> <=> 0 } )
 -->
     (MODIFY <CSP> ^VAL (COMPUTE ( <P0> + <P1> + <P2> ) // 3 )))
```

In OPS5, the "P" is a keyword to denote a production rule, and the identifier "G:1" is the rule name. Each rule name has to be distinct, and G:1 is formed by using the built-in function genatom, which generates a distinct atom every time it is called. The next five lines are the conditions for the rule to trigger. The last line is the action to be taken when all five conditions are met.

In simple English, the first condition states that the state of the target system has to be in the derived parameter calculation mode. The second condition states that there exists a derived parameter frame instance (ave_D-MAX) whose value slot is known. The third to fifth conditions state that the components for calculating the ave_D-MAX are availabilities of the red_D-MAX, the green_D-MAX and the blue_D-MAX. The action is to calculate the average value of the red_D-MAX, the green_D-MAX and the blue_D-MAX to form the ave_D-MAX.

5.4.4 Production Rules for Generating Symptoms

In the fourth phase, besides generating the symptom description frame instances, production rules for generating symptoms are also constructed. The following is a typical example of such a production rule:

```
(P G:15
     (CSP ^ID AVE_D-MAX ^VAL { <P0> < 0 } )
     (CSP ^ID RED_D-MAX ^VAL { <P1> < <P0> } )
     (CSP ^ID BLUE_D-MAX ^VAL { <P2> < <P0> } )
 -->
     (MAKE SYMPTOM ^ID LOW BLUE D-MAX ^SYM_CODE ( GENATOM ) )))
```

In this example, the condition for the symptom is that the ave_D-max is less than zero while the red_D-max, and the blue_D-max are both less than the ave_D-max. The action is to generate a symptom of low green D-max.

5.4.5 Production Rules for Generating Possible Causes

Also in the fourth phase, production rules for generating possible causes are also constructed. The following is a typical example of such a production rule:

```
(P G:16
     (SYMPTOM ^ID LOW GREEN D-MAX ^SYM_CODE <SC>)
 -->
     (MAKE CAUSE ^CAUSE_CODE 45 ^AF 50 ^SYM_CODE <SC> )
     (MAKE CAUSE ^CAUSE_CODE 6 ^AF 40 ^SYM_CODE <SC> )
     (MAKE CAUSE ^CAUSE_CODE 24 ^AF 10 ^SYM_CODE <SC> ))
```

The meaning of this example is that when there is a symptom of low green D-max, then the possible causes are those that the indices 45, 6 and 24 are pointing to with the certainty factors of .50, .40 and .10, respectively.

5.5 Expert System Skeleton Module

The expert system skeleton module uses the frame instances and the production rules generated in the target knowledge base to help end-users solve photofinishing process control problems. This is done in the following four phases:

1. Acquire control strip data and calculate derived parameters.

2. Generate and merge possible causes followed by conflicting causes resolution.

3. Acquire additional data and update certainty factors.

4. Present the final diagnosis.

The first phase prompts an end-user for the control strip parameter values, and, after all the control strip parameter values are available, it uses the derived data production rules to calculate the derived data parameter values.

The second phase uses of the symptom condition production rules to find out all the symptoms in the process. Once all the symptoms in the process are found, the system uses the possible cause production rules to find the possible causes. Then it combines possible causes that have the same cause description to form new (higher) certainty factors and/or uses the conflicting cause-pair description frames to form new (lower) certainty factors.

The third phase uses the relevant additional data slot in the cause-solution description frame instances to prompt the end-user to enter the relevant additional data value. In the event that the user questions why the additional data is needed, it uses the linkage between symptom and possible causes frames to provide the answer. It then uses the standard value and tolerance slots of the additional data frame to update the certainty factors of the possible causes.

The last phase displays all the possible causes generated one by one in descending order. In the event that the end user questions how the system arrives at its conclusion, it uses the linkage between symptoms and possible causes frames to give the answer. If the end-user asks for the prescription for the problem solution, the system uses the prescription slot of the cause-solution description frames to give the advice. Figures 5.1 to 5.3 show a typical consultation session of a target expert system.

In this example, when the target expert system was invoked by entering "run e6", the system responded by requesting the twelve control strip parameters for Process E-6. After the twelve readings were gathered, the system started diagnosing.

The diagnosing process included derived parameters calculation, symptoms generation, possible causes generation, possible causes merging and conflicting causes resolution. These steps were not listed so as not to overburden the end-user with unnecessary details.

After all possible causes were generated and merged, and conflicts resolved, the system prompted the user to enter relevant additional data. The system first requested for color

Figure 5.1: A typical consultation session: Enter Control Strip Parameters

```
run e6

>>> Control Strip Parameters Gathering <<<

Enter RED_D-MAX: -10
Enter GREEN_D-MAX: -6
Enter BLUE_D-MAX: -48
Enter RED_COLOR: -9
Enter GREEN_COLOR: -7
Enter BLUE_COLOR: -30
Enter RED_SPEED: -1
Enter GREEN_SPEED: 0
Enter BLUE_SPEED: -14
Enter RED_D-MIN: -1
Enter GREEN_D-MIN: 1
Enter BLUE_D-MIN: 0

>>> Diagnosing <<<
```

Figure 5.2: A typical consultation session con't: Enter Additional Data

```
>>> relevant Additional Data Gathering <<<

enter Color Developer specific gravity: help

  enter the additional data or
  type <cr> if the data is not available
       why for explanation
       stop to terminate questioning

enter Color Developer specific gravity: why

  *** there is symptom of LOW CONTRAST ( 95 )
                       AND FAST PROCESS ( 5 )
  *** possible cause ( 96 ) : Color Developer overconcentrated

enter Color Developer specific gravity: 1.04

enter First Developer specific gravity: why

  *** there is symptom of FAST PROCESS ( 50 )
  *** possible cause ( 50 ) : First Developer overconcentrated

enter First Developer specific gravity: 1.034

enter First Developer replenishment rate:

Color Developer replenishment rate: stop
```

Figure 5.3: A typical consultation session con't: Final Diagnosis

```
>>> Diagnosis <<<

===== possible cause ( 88 ) : Color Developer pH low  >>> help

  type <cr> for next possible cause
       how for explanation
       end to terminate session  >>> how

  *** there is symptom of LOW BLUE D-MAX ( 60 )
                      AND BLUE COLOR BALANCE ( 70 )  >>>

===== possible cause ( 47 ) : Color Developer overconcentrated  >>> how

  *** there is symptom of LOW CONTRAST ( 95 )
                      AND FAST PROCESS ( 5 )
  *** with Color Developer specific gravity = 1.04
            (std = 1.035 +/- 0.1000000E-01 ) >>>

===== possible cause ( 28 ) : Color Developer underreplenished  >>>

===== possible cause ( 28 ) : First Developer overreplenished  >>> end

>>> End of Session <<<
```

developer specific gravity. Upon a help response, the system explained what the possible choices of action were. When the end-user asked why color developer specific gravity was needed, the system responded with the explanations.

Subsequently, the user entered 1.04 for color developer specific gravity, why and 1.034 for first developer specific gravity, not available for first developer replenishment rate, and stopped the additional gathering session when color developer replenishment rate was requested.

The system then presented the diagnosis with color developer pH low as the most likely cause (80 percent certainty). When the end-user asked how the conclusion was arrived, the system provided the explanations. The system then presented a few more possible causes and finally the consultation was terminated when the end-user entered "end".

Chapter 6

Conclusion

Generally, the use of an expert system building tool that has the required knowledge structure built into the tool greatly decreases the development time for building a target expert system. It also eliminates the necessity of having a knowledge engineer as mediator to extract the knowledge from the experts and convert it into a computer program.

This is certainly true in this expert system tool for building photofinishing process diagnostic systems. That is, it is much easier for a photofinishing process diagnostic expert to use the knowledge based editor than to learn LISP, OPS5 or other expert system tools to build a diagnostic expert system from scratch.

Needless to say, the current interface for the experts to build a target expert system through the knowledge base editor is very primitive and, in many ways, restrictive. To make this tool a product, the knowledge base editor has to be greatly enhanced to allow for slot editing rather than frame editing. That is, currently, if any modification is to be done to one slot of the frame, the whole frame has to be re-typed. A better alternative is a menu-driven screen editor which allows an expert system builder to modify any slot or any frame.

Often, an expert system builder may have to create instances of frames that are very similar to one another. Therefore, another good feature would be to provide a copy utility

to create new frame instances with contents being the same as an existing frame.

The approach taken here for giving prescription advice to the processing control problem is very crude. It "hardwires" the prescription for any given possible cause. In general, one might want to consider other factors such as the type of processor that is used, the tank volume, the mixing procedure for the chemicals, etc., before a prescription is suggested.

Thus, another outstanding feature that will be good to have is to allow expert system builders to define their own frame types and linkages among the frames to establish the value of some of the currently "hardwired" slots. Of course, there are trade-offs for doing this: in making the tool more flexible, it also makes the tool more difficult to use.

It is also noticed that this tool is somewhat slow. It takes a long time to convert the intermediate knowledge file into the target knowledge base. The main reason is the way OPS5 generates new production rules: it creates a new file for each new rule created. For even a small system, at least fifty rules are generated and fifty new files created.

The only exception to the speed problem is the process of creating the intermediate knowledge file based on the expert system builder's input through the knowledge base editor. The reason is that the knowledge base editor is written in C rather than OPS5.

Therefore, if this is to turn into a product, perhaps the whole system should be re-written in C instead of in OPS5. Of course, the main emphasis of this project is to test out the idea of constructing a domain specific expert system tool for building photofinishing process diagnostic systems and speed is only the secondary consideration throughout this project.

In conclusion, it seems that such a domain specific expert system tool is useful and feasible.

Bibliography

[ARC 85] Automated Reasoning Corp., *Micro IN-ATE Product Description,* Automated Reasoning Corp., New York, 1985.

[Bran 83] Ann Brandeis, *Color Processing and Printing,* Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1983.

[Brow 85] Lee Brownston, Robert Farrell, Elaine Kant and Nancy Martin, *Programming Expert Systems in OPS5: An Introduction to Rule-Based Programming,* Addision-Wesley, Reading, Massachusetts 1985.

[Choi 86] King F. Choi, "An Expert System on E-6 Processing Diagnostics," *Technical Report* No. 222578X, Eastman Kodak Company, Rochester, New York, September, 1986.

[EK 80] Eastman Kodak Company, *Kodak Color Films,* eighth edition, Rochester, New York, 1980.

[EK 81] Eastman Kodak Company, *Z-119 Using Process E-6,* second edition, Rochester, New York, 1981.

[Glaf 58] Pierre Glafkides, translated by Keith M. Hornsby, *Photographic Chemistry,* vol. 1, Fountain Press, London, 1958.

[Gene 83] M.R. Genesereth, "The DART project," *Artificial Intelligence,* vol. 24, pp. 411-436, 1984.

[Gilm 85] John F. Gilmore and Kirt Pulaski, "A Survey of Expert Systems Tools," *IEEE Proceedings of the 2nd Conference on Artificial Intelligence Applications,* Miami Beach, Florida, December, 1985.

[Gilm 86] John F. Gilmore, Kirt Pulaski and Chuck Howard, "A Comprehensive Evaluation of Expert System Tools," *SPIE Applications of Artificial Intelligence III,* Orlando, Floride, April, 1986.

[Harm 86] Paul Harmon, *Expert Systems Strategies*, vol. 2, no. 8, August, 1986.

[Jack 76] J. Edward Jackson, "Statistics in Processing Control," *Journal of Applied Photographic Engineering,* vol. 2, no. 4, Fall, 1978.

[John 63] D. H. O. John and G. T. L. Field, *Photographic Chemistry,* Reinhold Publishing Corporation, New York, 1963.

[Kahn 86] Gary Kahn and John McDermott, "The Mud System," *IEEE Expert,* vol. 1, no. 1, pp. 23-32, Spring, 1986.

[Mich 82] R. S. Michalski, J. H. Davis, V. S. Bisht and J. B. Sinclair, "PLANT/ds: an expert consulting system for the diagnosis of soybean disease," *Proceedings of the Fifth European Conference on Artificial Intelligence,* Orsay, France, July, 1982.

[Shor 76] E. Shortliffe, *Computer-Based Medical Consultation: Mycin,* Elsevier, New York, 1976.

[Spen 75] D. A. Spencers, *Color Photography in Practice,* Focal Press Limited, England, 1975.

[Thom 73] W. L. Thomas (editor), *Handbook of Photographic Science and Engineering,* John Wiley and Sons, Inc., New York, 1973.

[Wate 86] Donald A. Waterman, *A Guide to Expert Systems,* Addision-Wesley, Reading, Massachusetts, 1986.