ANL/RE/RP--89482--V012
DRAFT

PRODIAG

Combined Expert System/Neural Network
for Process Fault Diagnosis

Volume 2: CODE MANUAL

J. Reifman
T. Y. C. Wei

September 1995

Argonne National Laboratory
9700 South Cass Avenue
Argonne, Illinois 60439

MASTER

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

OK

DISCLAIMER

**Portions of this document may be illegible
in electronic image products. Images are
produced from the best available original
document.**

© COPYRIGHT 1994 THE UNIVERSITY OF CHICAGO

No part of this work may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying and recording, or by any information storage or retrieval system, except as may be expressly permitted by the Copyright Law of the United States or in writing by Argonne National Laboratory.

Licensing inquiries should be directed to the Industrial Technology Development Center at Argonne National Laboratory, 1-800-627-2596.

Portions of this material resulted from work developed under a U.S. Government contract and are subject to the following license: the Government is granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable worldwide license in this material to reproduce, prepare derivative works, and perform publicly and display publicly. Any use of this material for other than Government purposes is outside of the scope of this license.

DISCLAIMER

THIS PROGRAM WAS PREPARED AS AN ACCOUNT OF WORK SPONSORED BY AN AGENCY OF THE UNITED STATES GOVERNMENT. NEITHER THE UNITED STATES GOVERNMENT NOR ANY AGENCY THEREOF, NOR THE UNIVERSITY OF CHICAGO, NOR ANY OF THEIR EMPLOYEES OR OFFICERS, MAKES ANY WARRANTY, EXPRESS OR IMPLIED, OR ASSUMES ANY LEGAL LIABILITY OR RESPONSIBILITY FOR THE ACCURACY, COMPLETENESS, OR USEFULNESS OF ANY INFORMATION, APPARATUS, PRODUCT, OR PROCESS DISCLOSED, OR REPRESENTS THAT ITS USE WOULD NOT INFRINGE PRIVATELY OWNED RIGHTS. REFERENCE HEREIN TO ANY SPECIFIC COMMERCIAL PRODUCT, PROCESS, OR SERVICE BY TRADE NAME, TRADEMARK, MANUFACTURER, OR OTHERWISE, DOES NOT NECESSARILY CONSTITUTE OR IMPLY ITS ENDORSEMENT, RECOMMENDATION, OR FAVORING BY THE UNITED STATES GOVERNMENT OR ANY AGENCY THEREOF. THE VIEW AND OPINIONS OF AUTHORS EXPRESSED HEREIN DO NOT NECESSARILY STATE OR REFLECT THOSE OF THE UNITED STATES GOVERNMENT OR ANY AGENCY THEREOF.

The Government reserves for itself and others acting on its behalf a royalty free, nonexclusive, irrevocable, world-wide license for governmental purposes to publish, distribute, translate, duplicate, exhibit, and perform any such data copyrighted by the contractor.

TABLE OF CONTENTS

LIST OF TABLES	iii
LIST OF FIGURES	v
GLOSSARY OF ACRONYMS	vi
NOMENCLATURE	vii
TERMINOLOGY	x
1.0 INTRODUCTION	1-1
References	1-3
2.0 THE EXPERT SYSTEM	2-1
2.1 Overall Expert System Architecture	2-1
2.2 Input Data Files	2-3
2.2.1 Plant Parameter File	2-3
2.2.2 Plant Information File	2-5
2.3 Output Data	2-7
2.3.1 User Interactive Window	2-7
2.3.2 Graphical Interface File	2-7
2.4 Reasoning Framework	2-8
2.4.1 Logic Flow	2-9
2.4.2 Time Window Selector	2-20
2.4.3 Major Routines	2-35
2.5 Knowledge Bases	2-39
2.5.1 Physical Rules Database	2-40
2.5.2 Component Classification Dictionary	2-49
2.5.3 Piping and Instrumentation Database	2-49
References	2-54
3.0 THE GRAPHICAL DIAGNOSTIC INTERFACE	3-1
3.1 Overall Architecture of the Graphical Interface	3-1
3.2 Graphics Customization	3-4
3.3 Graphical Interface	3-6
References	3-10
4.0 SAMPLE PROBLEM	4-1
References	4-2

LIST OF TABLES

Table 2.1.	Theoretical and Applied Time Coupling Between Thermal-Hydraulic Variables Based on Natural Feedback	2-21
Table 2.2.	Actual Trend and Trend to Be Used at the Opening of the Window $t_i = t_{open}(TW)$ for the $\{p,w\}$ and $\{p,\ell\}$ Classes of Variables	2-23
Table 2.3.	Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window is Open $t_{open}(TW2) < t_i < t_{open}(TW2) + \Delta t(TW2)$ for the $\{p,w\}$ Class of Variables	2-24
Table 2.4.	Previously Used Trend, Actual Trend, and Trend to Be Used at the Closing of the Time Window $t_i = t_{open}(TW2) + \Delta t(TW2)$ for the $\{p,w\}$ Class of Variables	2-25
Table 2.5.	Previously Used Trend, Actual Trend, and Trend to Be Used After the Time Window Has Been Closed $t_i > t_{open}(TW) + \Delta t(TW)$ for the $\{p,w\}$ and $\{p,\ell\}$ Classes of Variables	2-26
Table 2.6.	Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window is Open $t_{open}(TW3) < t_i < t_{open}(TW3) + \Delta t(TW3)$ for the $\{p,\ell\}$ Class of Variables	2-26
Table 2.7.	Previously Used Trend, Actual Trend, and Trend to Be Used at the Closing of the Time Window $t_i = t_{open}(TW3) + \Delta t(TW3)$ for the $\{p,\ell\}$ Class of Variables	2-27
Table 2.8.	Actual Trend and Trend to Be Used at the Opening of the Time Window $t_i = t_{open}(TW)$ for the $\{T\}$ and $\{\ell\}$ Classes of Variables	2-28
Table 2.9.	Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window Has Been Opened $t_i > t_{open}(TW)$ for the $\{T\}$ and $\{\ell\}$ Classes of Variables	2-28
Table 2.10.	Slop Times and Thresholds for the Various Classes of Variables	2-34
Table 2.11.	Instrumentation Error Rules (Module 00.00)	2-43
Table 2.12.	Q Rules with Constant Trend for Non-Separated Volumes (Module 10.00)	2-44
Table 2.13.	Q Rules with Varying Trend for Non-Separated Volumes in Open and Closed Loops (Module 20.00)	2-44
Table 2.14.	Q Rules with Varying Trend for Non-Separated Volumes in Closed Loops (Module 30.00)	2-45
Table 2.15.	Q Rules for Heat Exchangers (Modules 40.00 and 50.00)	2-45

LIST OF TABLES (Cont'd)

Table 2.16. Q Rules for Open Valves, Filters, Seals and Pumps (Module 120.00)	2-46
Table 2.17. Q Rules with Constant Trend for Separated Volumes (Module 100.00)	2-46
Table 2.18. Q Rules with Varying Trend for Separated Volumes (Module 90.00)	2-47
Table 2.19. CV Rules for Specific Components (Modules 60.00, 70.00 and 80.00)	2-48
Table 2.20. CV Rules for Separated Volumes (Module 110.00)	2-49
Table 2.21. Classification of Generic Non-Separated Volume Component Types as a Source or Sink of Mass, Momentum or Energy	2-50
Table 2.22. Prolog Predicates Used to Describe the Piping and Instrumentation Diagram of the Thermal-Hydraulic Process to Be Diagnosed	2-51
Table 3.1. Structure of the "graph_#" Files which Serve as the Means of Communication Between the Expert System and the Graphical Diagnostics Interface	3-9

LIST OF FIGURES

Fig. 1.1.	Schematic of the Overall Architecture of the Expert System and Graphical Diagnostics Interface	1-2
Fig. 2.1.	Schematic of the Overall Architecture of the Expert System and Names of the Various Files Necessary to Run the System	2-2
Fig. 2.2.	Detailed Flowchart of the Reasoning Framework of the Expert System	2-10
Fig. 2.3.	Detailed Flowchart for the Time-Window Selector Module	2-30
Fig. 3.1.	Schematic of the Graphical Diagnostic Interface and Names of the Various Files Necessary to Run the System	3-2
Fig. 3.2.	Complete File Structure Necessary for Running the Graphical Diagnostics Interface	3-3
Fig. 3.3.	Black and White Scanned Background Image for the Braidwood Chemical and Volume Control System	3-5
Fig. 3.4.	Background Image Overlaid with Static and Dynamic Objects Used for Diagnostics	3-7
Fig. 3.5.	Background Image Overlaid with Static Objects Indicating the Transient Locations	3-8
Fig. 4.1.	The Two Windows and Corresponding Commands Needed to Run the Expert System and the Graphical Diagnostics Interface	4-3
Fig. 4.2.	The Graphical Diagnostic Window and Textual Diagnostic Window Automatically Generated by the Graphical Diagnostic Interface Program and Used to Display the Diagnosis	4-4
Fig. 4.3.	Expert System Query Used to Control the Diagnostic Session	4-5
Fig. 4.4.	Expert System Screen After the Diagnostics is Initiated By Typing "go." at the First Query	4-6
Fig. 4.5.	Expert System Screen After Three Seconds of Transient Diagnostics	4-7
Fig. 4.6.	Graphical Diagnostic Display at 5.0 Seconds into the Transient	4-8
Fig. 4.7.	Graphical Diagnostic Display at 9.0 Seconds into the Transient	4-9
Fig. 4.8.	Textual Diagnostic Display for the First 9.0 Seconds of Transient ...	4-10

GLOSSARY OF ACRONYMS

CCD	-	Component Classification Dictionary
CRADA	-	Collaborative Research and Development Agreement
CRT	-	Cathode Ray Tube
CVCS	-	Chemical and Volume Control System
EOF	-	End of File
ES	-	Expert System
GDI	-	Graphical Diagnostics Interface
GUI	-	Graphical User Interface
HTX	-	Heat Exchanger
ID	-	Identification
NN	-	Neural Network
P&ID	-	Piping and Instrumentation Diagram
PID	-	Piping and Instrumentation Database
PRD	-	Physical Rules Database
RA	-	Reactor Analysis Division
RE	-	Reactor Engineering Division
ST	-	Signal Trend
T-H	-	Thermal-Hydraulic
TIFF	-	Tagged Image File Format
TW	-	Time Window
TWS	-	Time Window Selector

NOMENCLATURE

$\forall$	-	for all
-	-	normal or unchanging trend
/	-	logical negation
/-	-	nor normal or changing trend
$\uparrow$	-	increasing trend
$\nmid\uparrow$	-	not increasing trend
$\downarrow$	-	decreasing trend
$\nmid\downarrow$	-	not decreasing trend
?	-	unknown trend
Δp	-	pressure difference across a component
Δt	-	slop time
$\Delta t(TW2)$	-	slop time for TW2
$\Delta t(TW3)$	-	slop time for TW3
ϵ	-	threshold
ϵ_p	-	primary threshold
ϵ_s	-	secondary threshold
h	-	specific enthalpy
h_i	-	inlet specific enthalpy
h_o	-	outlet specific enthalpy
ℓ	-	level of separated-volume components
p	-	pressure
p_g	-	pressure of the gas phase in separated volumes
p_i or p_{up}	-	inlet or upstream pressure of a control volume
p_o or p_{down}	-	outlet or downstream pressure of a control volume

NOMENCLATURE (Cont'd)

Q_{mass}	-	mass imbalance indicator
Q_{mom}	-	momentum imbalance indicator
Q_{eng}	-	energy imbalance indicator
T	-	temperature
t_i	-	i-th time step into the transient
T_i or T_{up}	-	inlet or upstream temperature of a control volume
$T_{i,\text{cold}}$	-	heat exchanger cold-leg inlet temperature
$T_{i,\text{hot}}$	-	heat exchanger hot-leg inlet temperature
T_ℓ	-	temperature of the liquid phase in separated volumes
T_o or T_{down}	-	outlet or downstream temperature of a control volume
$T_{o,\text{cold}}$	-	heat exchanger cold-leg outlet temperature
$T_{o,\text{hot}}$	-	heat exchanger hot-leg outlet temperature
t_{open}	-	time during the transient at which the time window opens
$t_{\text{open}}(\text{TW1})$	-	time during the transient at which the general time window opens
$t_{\text{open}}(\text{TW2})$	-	time during the transient at which the time window TW2 opens
$t_{\text{open}}(\text{TW3})$	-	time during the transient at which the time window TW3 opens for a particular separated-volume component
TW1	-	general time window
TW2	-	time window for the $\{p,w\}$ class of variables in the entire piping and instrumentation diagram
TW3	-	time window for the $\{p,\ell\}$ class of variables for a specific separated-volume component that have both p and ℓ measurements
TW4	-	time window for the $\{T\}$ class of variables in the entire piping and instrumentation diagram
TW5	-	time window for the $\{\ell\}$ class of variables for all separated-volume components
w	-	mass flow rate

NOMENCLATURE (Cont'd)

w_a	-	average mass flow rate through a heat exchanger
w_{cold}	-	average mass flow rate through the cold-leg of a heat exchanger
w_{hot}	-	average mass flow rate through the hot-leg of a heat exchanger
w_i or w_{up}	-	inlet or upstream mass flow rate of a control volume
w_o or w_{down}	-	outlet or downstream mass flow rate of a control volume
x_o	-	expected value of plant parameter x

TERMINOLOGY

.ptimage	- text file that contains the information to direct the graphics program to open the display window with the desired "*.pti" file. It also contains the path for the directory where the graphics program is going to run
*.pti	- text file that holds the description of the objects overlaid onto the raw black-and-white background image of a *.tif file
*.tif	- compressed bit-map file of the black-and-white background image that is displayed on the graphics window
ccd.pl	- file containing the component classification dictionary
closed loop	- thermal-hydraulic loop with the same component at the two end conditions (see loop)
crada2.pti	- *.pti file that contains the information about the Braidwood Simulator Chemical and Volume Control System and is used as the graphical diagnostics interface (see *.pti)
crada2.tif	- *.tif file that contains the black-and-white background image of the Braidwood Simulator Chemical and Volume Control System (see *.tif)
crada3.pti	- *.pti file that contains the information about the location of the twenty component failures of the Braidwood Simulator Chemical and Volume Control System (see *.pti)
CV rules	- first-principle rules that infer the trend of unmeasured plant parameters
display	- script used to run the graphical diagnostics interface
dynamic objects	- display objects whose attributes may be updated during the diagnostics
es.scale2.0.f	- FORTRAN 77 source code used to process the raw transient data. The input to the code is the raw data file and the output are files input.data and input.data2
foton	- name of a Sun Sparc 2 workstation of the Reactor Engineering Division network
graph_#	- see graphical interface file

TERMINOLOGY (Cont'd)

graphical interface file	- is the ASCII file named "graph_#" generated by the expert system and read by the graphical diagnostics interface program. This file contains the necessary information to update the graphical interface
input.data	- input file for the expert system containing the normalized signal measurements at each time step of the transient
input.data2	- input file for the expert system containing plant-dependent information
loop	- continuous fluid circuit of monodirectional incompressible flow between two end conditions
negative Q logic rules	- first-principles rules that use one flow indication and one pressure indication and infer imbalance in both mass and momentum inventories
non-separated volumes	- components where the liquid and gas phases are not separated or that contain single-phase fluid
open loop	- thermal-hydraulic loop with two different components at the two end conditions (see loop)
pid.in	- Quintus Prolog executable code that automatically generates the piping and instrumentation database
pid.in.pl	- Quintus Prolog source code that automatically generates the piping and instrumentation database
pid.pl	- file containing the piping and instrumentation database that is automatically generated as the result of running "pid.in"
pid2.0.pl	- file containing the piping and instrumentation database
primary threshold	- numerical value used to determine the trend (increasing, decreasing, normal) of the plant parameters. For a given plant parameter the numerical value of the primary threshold is higher than the value for the secondary threshold
prodiag2	- expert system executive program written in Quintus Prolog
prodiag2.0.pl	- expert system source code written in Quintus Prolog
Prolog	- computer language meaning programming in logic

TERMINOLOGY (Cont'd)

ptimage	-	executable C program for the graphics
ptimage.res	-	text file used to initialize the background color, foreground color, etc. of the graphical window
ptimage.uid	-	binary configuration file for the graphics program
q_rules2.pl	-	file containing the physical rules database
Q rules	-	first-principles rules that infer the type and trend of mass, momentum and energy imbalances in a control volume
Q value	-	mass, momentum or energy imbalance indicator
secondary threshold	-	numerical value used to determine the trend (increasing, decreasing, normal) of the plant parameters. For a given plant parameter, the numerical value of the secondary threshold is lower than the value for the primary threshold
separated-volumes	-	components where the liquid and gas phases are separated
slop time	-	time period for which the time window is active or open
static objects	-	display objects that remain unchanged throughout the diagnostics
time window	-	is an abstraction used to represent the time period in which the change in the variable trends are associated with their thermal-hydraulic coupling due to instrument delay and feedback effects
user interactive window	-	UNIX window where the expert system is run

1.0 INTRODUCTION

We recommend the reader first review Volume 1 of this document, Code Theory, before reading Volume 2. In this volume we make extensive use of terms and concepts described and defined in Volume 1 which are not redefined here to the same extent. To try to reduce the amount of redundant information, we have restricted this volume to the presentation of the expert system code and refer back to the theory described in Volume 1 when necessary. Verification and validation of the results are presented in Volume 3, Application, of this document. Volume 3 also presents the implementation of the component characteristics diagnostic approach through artificial neural networks discussed in Volume 1. We decided to present the component characteristics approach in Volume 3, as opposed to write a separate code manual for it, because the approach, although general, requires a case-by-case analysis.

The purpose of this volume is to present the details of the expert system (ES) portion of the PRODIAG process diagnostic program. In addition, we present here the graphical diagnostics interface (GDI) and illustrate the combined use of the ES and GDI with a sample problem. For completeness, we provide the file names of all files, programs and major subroutines of these two systems, ES and GDI, and their corresponding location in the Reactor Analysis Division (RA) computer network and Reactor Engineering Division (RE) computer network as of 30 September 1995.

The diagnostic system was developed to run on a UNIX platform under X-windows with SunOS 4.1.3 operating system and a color monitor. The ES was implemented in Quintus Prolog [1.1] and the GDI was implemented in C.

Figure 1.1 shows a schematic of the combined ES/GDI system which runs in an interactive mode. At each time step, the user may either continue the diagnostics or stop it by typing the appropriate command in the user interactive window. Two input data files containing the plant parameter values and plant specific data are input to the ES. The file containing plant parameter data emulates the plant signals that would be input to the ES once it is deployed. Given the plant data at every time step, the reasoning framework of the ES drives the system with the use of three knowledge bases, (physical rules database, component classification dictionary and piping and instrumentation database), to hypothesize possible faulty component candidates. The diagnosis is sent to the user interactive window, where the user can either continue or stop the diagnostics session, and

Fig. 1.1. Schematic of the Overall Architecture of the Expert System and Graphical Diagnostics Interface

to a graphical interface file. The graphical interface file is input to the GDI program which modifies the status of the graphical objects and displays the updated graph and textual diagnosis on the computer color monitor.

In the following chapters we describe, in detail, each one of the components of the ES/GDI system depicted in Fig. 1.1 and illustrate its application with a sample problem. For that, the remainder of this volume is divided into three chapters:

- 2.0 The Expert System
- 3.0 The Graphical Diagnostics Interface
- 4.0 Sample Problem

The ES is documented in Chapter 2. Section 2.1 describes the overall architecture of the ES including the file names and locations of the programs and subroutines necessary to run the ES. The input data files for the ES are presented in Section 2.2 and the output data are presented in Section 2.3. In Section 2.4 we describe the reasoning framework of the ES including the logic flow and the major routines of the driving program. In Section 2.5 we present the three distinct knowledge bases of the ES. The GDI is documented in Chapter 3. Section 3.1 describes the overall architecture of the GDI including the file names and locations of the programs and files necessary to run the GDI. The graphics customization is presented in Section 3.2, followed by the description of the graphics display and textual diagnostic information in Section 3.3. A sample problem illustrating the ES/GDI system is presented in Chapter 4.

REFERENCES

- [1.1] Quintus Corporation, Palo Alto, California, Release 3, 1991.

2.0 THE EXPERT SYSTEM

In this chapter we present the details of the ES portion of the PRODIAG process diagnostic program. For completeness, we provide the file names of all programs, input files and major subroutines necessary for developing and running the ES as well as their location in the RE computer network. The entire set of files of the ES are located in the directory `/home/foton1/doa/weicrada/prolog.1` on the Sun Sparc 2 "foton" workstation of the RE network. For instance, the full path for the ES executable program, "prodiag2", is `/home/foton1/doa/weicrada/prolog.1/prodiag2`. For simplicity, in this and the following chapters we refer to this executable program as "prodiag2" knowing that it is located in the aforementioned directory.

The ES was developed to run on a UNIX platform under X-windows with the SunOS 4.1.3 operating system and a color monitor. The ES was implemented in Quintus Prolog [2.1] with the development software licensed to the Sun Sparc 2 "foton" workstation.

For a more comprehensive definition of the terms and concepts presented in this section, the user may need to refer back to Volume 1 of this document, Code Theory.

2.1 Overall Expert System Architecture

In this section we present the overall architecture of the ES including the corresponding files containing the input data, executable code, knowledge bases and output data. In the remaining sections of this chapter we describe each one of these files in detail.

Figure 2.1 shows a schematic of the overall architecture of the ES and names of the various files necessary to run the system. The ES requires two input files. The first one, named "input.data", is read at every time step and contains the transient time associated with each time step and the normalized values of the corresponding plant parameters, i.e., the transient time history. The second one, named "input.data2", is read only once and contains plant-dependent information such as the generic type of each component, component fluid type and threshold values for each plant parameter. This input data file complements the information contained in the third knowledge base, the Piping and Instrumentation Database, of the ES. These two input files are described in more detail in Section 2.2.

Fig. 2.1. Schematic of the Overall Architecture of the Expert System and Names of the Various Files Necessary to Run the System

The executable Prolog file for the ES is termed "prodiag2". The ES is run interactively in the user interactive window by first going to the directory where the file resides, /home/foton1/doa/weicrada/prolog.1, and typing "prodiag2" at the operating system prompt and pressing the <return> key. The program will be loaded and will query the user to enter the word "go." (do not forget the dot "." after the word go) to start or continue execution, or "stop." to terminate the diagnostic session. After termination of the diagnostic session, the user can get out of Prolog and back to the operating system prompt by typing control-d. If desired, the ES can easily be modified to run continuously until the end of the data file or until the diagnosis is interrupted. In Chapter 4 we provide a sample problem describing the execution of the ES. The source code for the executable "prodiag2" is termed "prodiag2.0.pl". Note that all Prolog files are suffixed with ".pl". The reasoning framework coded in "prodiag2.0.pl" is presented in Section 2.4 below.

The ES contains three distinct knowledge bases, the Physical Rules Database (PRD), the Component Classification Dictionary (CCD) and the Piping and Instrumentation Database (PID). The PRD is located in the Prolog file termed "q_rules2.pl". This file contains both the first-principles rules that indicate the type and trend of the imbalance, i.e., the Q rules, as well as the rules that indicate the trend of unmeasured plant parameters, i.e., the CV rules. It also contains a few instrumentation rules used to detect instrumentation error. The CCD is located in the Prolog file "ccd.pl" and the PID is located in a file termed "pid2.0.pl". These three knowledge bases are described in detail in Section 2.5.

The ES output is sent to the user interactive window under Prolog and to a file termed "graph_#" used by the GDI program. The user interactive window is the UNIX window where the executable "prodiag2" runs. The file termed "graph_#" is generated by the ES after the diagnosis of each time step and used by the graphics program to update the status of the dynamic objects at each time step. The output data is presented in greater detail in Section 2.3 and Chapter 3 below.

2.2 Input Data Files

In this section we describe the two input data files, "input.data" and "input.data2", depicted in Fig. 2.1 that are necessary to run the ES. Both files should be located in the same directory as the executable file "prodiag2", that is, /home/foton1/doa/weicrada/prolog.1.

2.2.1 Plant Parameter File

The input data file "input.data" is read by the ES at each time step or sampling time for which the ES is activated. This file emulates the information that would be obtained directly from the plant if the ES were running on-line. File "input.data" contains the transient time, e.g., at 1 second intervals starting at 1 second, followed by the normalized values of the plant parameters at the associated time step.

File "input.data" is generated by processing raw transient data with a FORTRAN program termed "es.scale2.0.f". In our case, the raw transient data was generated via the full-scope Braidwood simulator. The "es.scale2.0.f" program is specific to a given thermal-hydraulic

(T-H) system and needs to be modified for every T-H system since it depends on the system piping and instrumentation diagram (P&ID). This program is located in the directory /home/foton1/doa/weicrada/norm.braidwood. The raw transient data files associated with the transient simulations are also located in this directory and are named "cvXX.YY.A", where "XX" represents the failure type, "YY" represents the failure severity and the suffix "A" indicates that the file is a "raw" transient file. For instance, file "cv07.100.A" contains the raw transient data for transient event cv07 (clogging of seal injection filter) at 100% severity level.

The processed files are also located in this directory and are named similar to the raw files but with a suffix "D" instead of "A". For instance, file "cv07.100.D" is the processed version of "cv07.100.A". The processed files are renamed "input.data" when copied to the directory /home/foton1/doa/weicrada/prolog.1, since "input.data" is the file name that the ES code "prodiag2" looks for. The data files for the blind test are located in the directory /home/foton1/doa/weicrada/blind.braidwood, where the raw files are named "argonne.#.A" and the processed files are name "argonne.#.E". Here, "#" is the file identification number and runs from 1 to 44 with the exception of numbers 42 and 43.

As discussed in the foregoing paragraphs, file "input.data" contains the transient time followed by the normalized values of the plant parameters at the associated time step. At each time step, a "value" is associated with each one of the twelve possible plant parameter measurements (input temperature, average temperature, outlet temperature, input pressure, average pressure, output pressure, level 1, level 2, pressure difference, input flow, average flow and output flow) of a component. Level 1 and level 2 indicate two redundant level signals measured by two different instruments in tanks with separated gas and liquid volumes. The pressure difference indicates the Δp across a component. The "values" are for each component of each loop following the loop and component sequences used in the PID file "pid2.0.pl" [2.2]. For each and every component of each T-H loop, the twelve plant parameter "values" of the input temperature, average temperature, outlet temperature, input pressure, average pressure, output pressure, level 1, level 2, pressure difference, input flow, average flow and output flow, are presented. Since, in general, we do not have instruments that measure these plant parameters for each component, the "values" of nonexistent instruments are set to "none." in the "input.data" file. The word "none." is used as a flag for the ES to identify the lack of instrumentation for the particular plant parameter.

2.2.2 Plant Information File

The input data file "input.data2" contains plant-dependent information that is complementary to the PID file "pid2.0.pl" [2.2]. This file is read by the ES only once at the beginning of the system execution. File "input.data2" is automatically generated in the appropriate format when the FORTRAN program termed "es.scale2.0.f" located in the directory /home/foton1/doa/weicrada/norm.braidwood is used to generate the plant parameter file "input.data". An "input.data2" file is generated each time a raw transient data file is processed. However, all generated "input.data2" files are identical and only one copy of this file should be placed in the directory /home/foton1/doa/weicrada/prolog.1 where the ES is executed. The "es.scale2.0.f" program needs to be modified to generate the appropriate "input.data2" file for each T-H system.

File "input.data2" contains some (twenty-nine) of the attributes of the plant components of the P&ID and is complementary to the PID file "pid2.0.pl". At the beginning of the system execution, the ES reads file "input.data2" and fills in the empty slots of the component attribute values of the PID file "pid2.0.pl" with the read values. By providing some plant-dependent component data through the "input.data2" file, we simplify and speed-up the user-query-based construction of the "pid2.0.pl" file discussed in Section 2.5.3. Below we describe, in detail, how the user assembles an input for "input.data2" from scratch.

File "input.data2" contains the values of twenty-nine attributes for each and every component of each T-H loop in the P&ID. The sequence of components and loops is the same one used in the PID file "pid2.0.pl". If the entire P&ID consisted of only two components, an open valve and a tank, the "input.data2" file would be as follows:

open_valve.	sp_liquid.	not_sep.	cont_bld.	pos.	
0.025.	0.025.	0.025.	0.0025.	0.0025.	0.0025.
0.05.	0.05.	0.05.	0.004.	0.004.	0.004.
0.025.	0.025.	0.0025.	0.0025.	0.05.	0.004.
0.05.	0.05.	0.05.	0.004.	0.004.	0.004.
tank.	non_cond_gas_liq.	not_sep.	cont_bld.	pos.	
0.025.	0.025.	0.025.	0.0025.	0.0025.	0.0025.
0.05.	0.025.	0.05.	0.004.	0.004.	0.004.
0.025.	0.025.	0.004.	0.004.	0.05.	0.004.
0.05.	0.05.	0.05.	0.004.	0.004.	0.004.

Every five lines of the file corresponds to a different component. Hence, the first five lines represent the twenty-nine attribute values of the open valve and the next five lines represent the twenty-nine attributes of the tank. The first five attributes of a component correspond to the component generic type, fluid type, control volume type, name of surrounding system and relative pressure information, respectively, that characterize a plant component. Generic components types are the basic components that make up the P&ID of T-H systems. For instance, open valve (entered in "input.data2" as "open_valve"), closed valve (entered as "closed_valve"), junction, pump, seal, filter, demineralizer, pipe, heat exchanger (entered as "heat_exchanger"), turbine, heater, tank and pressurizer. Each generic component type is classified in the CCD database (see Section 2.5.2 below and Section 3.3 in Volume 1) as a source or sink of mass, momentum, or energy. The current version of PRODIAG allows for two types of fluid, single-liquid and non-condensable gas over single-phase liquid. These two fluid types are entered in "input.data2" as "sp_liquid" and "non_cond_gas_liq", respectively. There are two types of control volumes, separated and non-separated, that should be entered in "input.data2" as "separated" and "not_sep", respectively. Name of surrounding system is the name of the system that describes the location of the component in the plant. For instance, auxiliary building, containment building, etc. This additional information is not used in the current version of PRODIAG, nevertheless, a name should be entered. Relative pressure information can have one of three values, "pos", "neg" or "pos_neg", indicating positive, negative or both positive and negative relative pressure in relation to the environment surrounding each component. The shell-side of a heat exchanger can have both positive and negative relative pressures. It could have a positive pressure relative to the tube-side and a negative pressure relative to the environment surrounding the shell-side of the heat exchanger.

The subsequent twenty-four attributes correspond to the values of the primary threshold ϵ_p and secondary threshold ϵ_s (see [2.3] and [2.4]) for the twelve plant parameters of each component. The sequence is the following: ϵ_p for inlet, average and outlet temperature; ϵ_s for inlet, average and outlet temperature; ϵ_p for inlet, average and outlet pressure; ϵ_s for inlet, average and outlet pressure; ϵ_p for level 1 and level 2; ϵ_s for level 1 and level 2; ϵ_p for pressure difference between inlet and outlet; ϵ_s for pressure difference between inlet and outlet; ϵ_p for the inlet, average and outlet flow; and ϵ_s for the inlet, average and outlet flow. Section 2.4.2 below provides guidelines for determining the values of ϵ_p and ϵ_s and Table 2.10 shows the values used for a particular application. File "input.data2" provides a numerical value for each one of these twenty-four component attributes even when an instrument does not exist.

2.3 Output Data

In this section we describe the two output streams, the user interactive window under Prolog and a file termed "graph_#", which are updated and generated, respectively, by the ES at each diagnostic time step.

2.3.1 User Interactive Window

The user interactive window is the window under Prolog where the executable code runs. That is, the window where the user types "prodiag2" to run the ES. Two types of messages are passed from the ES to the user at each diagnostic time step. In addition, at the end of each time step the system queries the user if the diagnostic session should be continued or stopped. To continue the user should type "go." (do not forget the dot "." after the word go), and to stop the user should type "stop." followed by a control-d to return to the system prompt. Therefore, this window provides the means for the user to terminate the diagnostic session. If desired, the ES can easily be modified to run continuously from the first time step in the "input.data" file until the end of the file or until the diagnosis is interrupted.

The first type of message that the ES passes to the user is the result of a set of write statements that the system developer can use during debugging, or that the system user can use to better follow the logic reasoning steps of the ES. The second type of message that the ES passes to the user through the user interactive window is the result of the diagnosis, i.e., the list of possible faulty component candidates. This is the same diagnosis information displayed in the text window of the GDI discussed in Section 3.3.

2.3.2 Graphical Interface File

At every diagnostic time step that the ES makes an inference it automatically generates an output text file termed "graph_#". The symbol "#" represents an integer number starting with the value of "1" for the first diagnostic time step and increasing by one at each subsequent time step. For instance, at the first time step the ES generates "graph_1", at the second diagnostic time step it generates file "graph_2", and so on.

The "graph_#" text files serve as the means of communication between the ES and the GDI. The "graph_#" files contain the status of the dynamic objects of the graphical window of the GDI that are updated at every diagnostic time step. The status of the dynamic objects includes the numerical value and color of the plant parameter sensors, transient time, and the color of the pipes and boxes surrounding the components. This communication text file between the ES and the GDI also contains the textual diagnostic information that is displayed in the text diagnostic window of the GDI.

Once the script for running the GDI is invoked (termed "display" and located in the directory ~reifman/weicrada/prolog.1) it keeps looking for "graph_1". Once the ES generates "graph_1" after the first diagnostic time step is completed, the GDI reads this file, updates the graphical and textual windows accordingly, deletes "graph_1" and starts looking for "graph_2". This sequence is repeated until the diagnostic session is terminated. This approach ensures synchronization between the ES results and the GDI. An in-depth description of "graph_#" is given in Section 3.3.

2.4 Reasoning Framework

In this section we present the reasoning framework of the ES including a logic flowchart of the code. We introduce the Time Window Selector module used to time correlate the plant parameter signals and present and explain the function of the key routines of the ES.

Embedded in both the diagnostic strategy (or reasoning framework) and diagnostic rules of the PRODIAG ES are two simplifications about the diagnostic problem. These simplifications are not limitations of the proposed concept, but rather assumptions made to simplify its implementation in this first attempt. It is possible, however, that in future work we may find it impractical or not possible to lift these two simplifications and rewrite the code without them.

The first simplification limits the scope of the diagnostic problem to *single faults*. The single-fault assumption is translated mainly into the mass and momentum conservation equation rules through the following effects. The failure of a component functionally classified as a source or sink of mass will cause *all* pressure sensors to *either* increase *or* decrease, while the trends in the flow sensors can *both* increase *and* decrease depending on

their relative position with respect to the location of the faulty component. Therefore, once a mass problem has been detected in a T-H loop and one sensor in the loop indicates a pressure decrease, by the single-fault assumption the diagnostic system can then infer that pressure is decreasing everywhere in the loop. On the other hand, the failure of a component functionally classified as a source or sink of momentum will cause *all* flow sensors to *either* increase *or* decrease, while the trends in the pressure sensors can *both* increase *and* decrease depending on their relative position to the location of the faulty component.

It is possible that the code will correctly diagnose multiple failures as long as all faulty components have the same functional classification as *either* a mass or a momentum component and that the failures cause *either* an increase or a decrease in mass or momentum inventories, respectively. It is also possible that the code will detect the *occurrence* of multiple failures (without being able to identify the faulty components) through inconsistencies in the inferences in the trend and type of the T-H loop imbalances [2.5]. Of course, the possibility of identifying a faulty condition that is outside of the scope of PRODIAG will depend, as the entire diagnostic approach does, on the existence of the right sensor types at the right locations.

The second simplification of the code limits the type and state of the fluid in the components of the system to *subcooled water* and *subcooled water/non-condensable gas* combinations. The properties of single-phase liquid water allow us to use the assumption that the fluid is incompressible which is embedded in the driving logic of the reasoning framework, the dynamic treatment of the diagnostic through the Time Window Selector module discussed in Section 2.4.2 [2.3, 2.4, 2.6 and 2.7], and the static first-principles diagnostic rules presented in Section 2.5.1 [2.8]. The code will also work for other fluids if their physical characteristics are similar enough to those of subcooled water.

2.4.1 Logic Flow

The overall diagnostic strategy of the PRODIAG ES is illustrated through a flowchart in Fig. 2.2. Since the current version of ES runs in an interactive manner, it requires the user's input (through a query) to continue or stop the diagnostic session at each time step.

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

Fig. 2.2 Detailed Flowchart of the Reasoning Framework of the Expert System (Cont'd)

The ES does, however, automatically stop if certain conditions are satisfied indicating the fact that the current plant conditions are outside the scope of the ES. At each time step, plant parameter data are read and their trend determined by the Time Window Selector module discussed in detail in Section 2.4.2 below. When the plant parameters are varying from their expected values the diagnosis is initiated by first prioritizing the loops of the P&ID to be searched, transporting enthalpy and constant flow through the entire P&ID, and then applying the first-principles rules of the Physical Rules Database presented in Section 2.5.1. The loops are prioritized based on the number of varying signals in the loop, where the loop with the largest number of varying signals is searched first. As discussed in Section 2.5.1 and depicted in the flowchart in Fig. 2.2, the rules are grouped in modules where the same module numbering scheme is preserved in the actual implementation of the code.

The ES strategy is one that maximizes the possibility of estimating the trends of unmeasured plant parameters (through the CV rules), which are then used to make inferences about the possible imbalances in the conservation of mass, energy, or momentum (through the Q rules). The proposed approach alleviates, to the maximum extent possible, the problem of limited instrumentation and hence increases the diagnostic resolution of the ES.

As illustrated in Fig. 2.2, this feature of the ES is achieved by iteratively applying the Q rules and CV rules for all prioritized loops. The first time the non-separated volume Q rules are applied (modules 10.00-30.00 presented in Tables 2.12-2.14), the objective is to use them to determine which components *are not* experiencing imbalances, i.e., which components have constant (-) Q_{mass} , Q_{eng} , and Q_{mom} . This information is then used by the non-separated volume CV rules through modules 60.00-80.00 presented in Table 2.19, to estimate the trends of unmeasured plant parameters in the non-separated volume components. The subsequent times the non-separated volume Q rules are applied, the objective is to use them to determine which components *are* experiencing imbalances. If separated volume components are present in the T-H loop, then a sequence of separated-volume Q rules and CV rules (modules 90.00-110.00 presented in Tables 2.18, 2.17, and 2.20) are also applied.

This iterative approach may be interrupted, automatically halting the diagnostic session, if instrumentation errors are identified, inconsistencies are found in the implementation of the Q rules, rules for specific non-separated volume components (heat exchangers) are

applicable, or specific separated volume component rules are activated. A limited set of first-principles rules, described in module 00.00 and presented in Table 2.11 in Section 2.5.1, has been designed to detect instrumentation failure in a few component types. Whenever any of these rules are activated, a sensor and not a component has failed and the diagnostic session is terminated. The diagnostic session will also be interrupted by inconsistencies in the inferences about the imbalances in T-H loops. Inconsistencies can be caused by the occurrence of multiple failures or undetected instrumentation errors. Inconsistencies are used as a flag to alert the user that the current state of the plant, indicated by the plant parameters values, is beyond the scope of the ES [2.5]. Finally, the iterative use of the first-principles rules can also be interrupted if rules designed to detect the failure of specific types of components (modules 40.00 and 90.00 presented in Tables 2.15 and 2.18, respectively) are activated indicating the malfunction of the component.

If the iterative approach is not interrupted by any of the reasons described in the foregoing paragraph after all prioritized loops have been searched, then enthalpy is once again transported in all loops of the P&ID and the possible faulty loops are re-prioritized. For a given loop, the instrument error rules of module 00.00 and the Q rules for non-separated volumes of modules 10.00-30.00 are once again applied. If the loop is found to have an increasing or decreasing imbalance in Q_{mass} , Q_{mom} or Q_{eng} , then the possible faulty component candidates are identified, the loop search is terminated, and the ES queries the user to continue the diagnostic session for the next time step or end the diagnostic session. An increasing or decreasing imbalance in Q_{mass} , Q_{mom} or Q_{eng} , in a loop defines a "unique" diagnosis. If a unique diagnosis can not be made in the loop (that is, either Q_{mass} , Q_{mom} or Q_{eng} , are not increasing or decreasing), the search continues through the next loop in the re-prioritized list of loops until a unique diagnosis is made. If after all loops have been searched a unique diagnosis is not made, then the rules using one pressure measurement and one flow measurement (i.e., the negative Q logic rules) are applied to infer the possibility of mass and momentum imbalances [2.9]. When both pressure and flow measurements are not available in a loop, the search continues in a final attempt to determine off-normal behavior in Q_{mass} and/or Q_{mom} . If a non-unique diagnosis is made, the possible faulty components are identified and the ES queries the user to continue the diagnostic session for the next time step or end the diagnostic session. If no diagnosis is made, a "no diagnosis" message is printed and the ES queries the user to continue or stop the diagnostic session. We should point out that when Q_{mass} imbalances are inferred and the imbalanced control volume includes junctions, then the search is continued in the loops across the "mass imbalanced" junctions.

2.4.2 Time Window Selector

As discussed in Chapter 2 of Volume 1, Dynamic Effects, it is necessary to account for the dynamic effects in the signal processing before the ES rules in the Physical Rules Database are utilized. This task is performed in the Time Window Selector (TWS) module of the ES alluded to in the previous section. The theory outlined in Chapter 2 of Volume 1 served as the basis for the development and implementation of the TWS module in PRODIAG [2.4]. In this section, we discuss the practical implementation of the theory, present approximations used to simplify the logic and show the basis for these approximations. The reader should refer to Chapter 2 of Volume 1 of this document for an in-depth theoretical basis of the TWS concept.

In the TWS module, the qualitative trend of the T-H variables is determined by taking into account the time constants associated with instrumentation time response and natural T-H feedback, and determining if the diagnostics session should be halted due to automatic control system action. We should point out, however, that the TWS has not been developed to account for noise in the signal data. It is assumed that the signal data have already been filtered by a low frequency bandpass filter to filter out noise in the data.

The development and implementation of the TWS module reflect the fact that the underlying objective of handling the dynamic effects in a process during a transient is to account for the instrumentation delay and natural feedback and disregard the effects of process automatic control actions. Control actions have the potential to mask signals or to drive signals from their normal steady-state values which, in turn, could incorrectly activate the diagnostic rules of the PRD and lead to incorrect inferences. On the other hand, natural feedback effects should be an integral part of the reasoning since they allow us to infer trends of parameter signals for which measurements are not available, and through that, to characterize imbalances which otherwise could not be detected. This can be done by solving the balance equations for the T-H variables [2.9], then using the calculated parameters to characterize imbalances in mass, energy, and momentum. For example, by taking into account the delayed response of a heat exchanger outlet temperature due to a flow change resulting from a mass or momentum problem in the loop, we can infer the trend of the flow through the heat exchanger which can then be applied to characterize the mass or momentum imbalances.

Before the trend of the signals to be used at each time step during the transient is inferred in the TWS module, the raw T-H signals are first processed in the Signal Trend (ST) module illustrated in Fig. 2.2. This module determines the *actual* trend of each variable in the entire P&ID of the process based on simple threshold logic. If the value of the variable is within $x_0 \pm \epsilon$, where x_0 is the expected value of x and ϵ is the threshold, the variable is said to be constant (-), otherwise the variable is said to be increasing ($\uparrow$) or decreasing ($\downarrow$). In the ST module, each variable is compared with two thresholds, the primary threshold ϵ_p used to inform the algorithm that the variable is varying enough to open the Time Window (TW) for the associated class of variables, and the secondary threshold ϵ_s used to inform the algorithm that the variable is varying enough to be considered in the analysis.

As discussed in the description of the theory of the TWS concept (see Chapter 2 of Volume 1 of this report), the dynamic effects in the signal processing are handled by grouping T-H variables into classes based on the type of control volume (separated volume or non-separated volume) where the signal is located and the T-H coupling characteristics [2.4]. After the T-H variables are grouped into classes, each class is treated independently and all variables within a class are treated in the same way through the TW concept. In essence, the TW concept is an abstraction that specifies a window in time in which the change in the variable trends are associated with their T-H coupling due to instrument delay and feedback effects. The length of the TW should be such that effects of the system control action would occur after the TW is closed. The five classes of coupled T-H variables, $\{p, w\}$, $\{h_i, h_o, w\}$, $\{p, \ell\}$, $\{\ell, w\}$ and $\{h_i, h_o, w\}$, considered in the implementation of the TWS module and discussed in the theory are duplicated here in Table 2.1.

Table 2.1. Theoretical and Applied Time Coupling Between Thermal-Hydraulic Variables Based on Natural Feedback

Control Volume	Non-separated		Separated		
Coupled Variables	$\{p, w\}$	$\{h_i, h_o, w\}$	$\{p, \ell\}$	$\{\ell, w\}$	$\{h_i, h_o, w\}$
Correlating Equation	mass and momentum balance	energy balance	gas equation of state	liquid mass balance	liquid energy balance
Time Coupling	fast	slow	fast	slow	slow
Approximate Coupling	$\{p, w\}$	$\{T\}$	$\{p, \ell\}$	$\{\ell\}$	$\{T\}$

The first major approximation in the implementation of the TW concept is the coupling of the T-H variables [2.4]. Here, the theoretical coupling illustrated in the first row of Table 2.1 is approximated by grouping the signals into four classes of variables, $\{p,w\}$, $\{p,\ell\}$, $\{T\}$, and $\{\ell\}$, presented in the last row of the table. For non-separated volumes, the $\{p,w\}$ coupling is treated just like in the theory; the same is also true for the $\{p,\ell\}$ coupling in separated volumes. However, the treatment of the $\{h_i, h_o, w\}$ coupling for either volume type is simplified here. Flow w is decoupled from enthalpy h and is not considered other than in the $\{p,w\}$ coupling. The inlet enthalpy h_i and outlet enthalpy h_o are not differentiated and are treated as temperature T . For separated volumes where p is not measured, the treatment of the $\{\ell, w\}$ coupling is also simplified where ℓ is decoupled from w and is the only variable treated. When p is also measured in addition to ℓ , then ℓ is treated through the $\{p,\ell\}$ coupling. When p is measured and ℓ is not measured, p is treated just like ℓ is in the $\{\ell, w\}$ coupling, hence this case is not discussed further. The basis and consequences of these approximations are discussed in the following paragraphs.

There are a total of five time windows, a general TW and one TW for each one of the four classes of variables [2.3]. The general TW is termed TW1, the TW for $\{p,w\}$ is termed TW2, the TW for $\{p,\ell\}$ is termed TW3, the TW for $\{T\}$ is termed TW4 and the TW for $\{\ell\}$ is termed TW5. The TW for a class of T-H variables opens when any of the variables of the class located anywhere in the P&ID reaches its primary threshold ϵ_p . The opening of the TW for a class of signals causes the general TW to open, which triggers the beginning of the diagnostics and allows the variables for which the corresponding TW is not open, to open if the corresponding secondary threshold ϵ_s is reached, or to be taken as constant or unchanging (-) otherwise. The openings of the four TWs for the four classes of variables are decoupled from each other, apart from their coupling through the general TW, and once opened they are treated separately.

Table 2.2 shows the actual trend and the inferred trend (or trend to be used) at the opening of TW2 for the $\{p,w\}$ class of variables. If the actual trend of the variable, determined in the ST module, is increasing ($\uparrow$) or decreasing ($\downarrow$), then the inferred trend at the opening of TW2 at time $t_i = t_{\text{open}}$ is also increasing or decreasing, respectively. However, if the actual trend is constant (-) at t_{open} , then we infer that the trend is unknown (?). The unknown (?) trend is an abstraction used to represent the fact that the trend of the variable should not be used for diagnostics. Once TW2 opens at t_{open} , the trends used for $\{p,w\}$ variables in subsequent time steps t_i , until the diagnosis is halted, depend on the three possible states of

Table 2.2. Actual Trend and Trend to Be Used at the Opening of the Window $t_i = t_{\text{open}}(\text{TW})$ for the $\{p, w\}$ and $\{p, \ell\}$ Classes of Variables

Actual Trend at $t_i = t_{\text{open}}$	Trend to be Used at $t_i = t_{\text{open}}$
↑	↑
↓	↓
-	?

TW2, open, closing and closed, the trend used at the previous time step t_{i-1} , and the actual trend at t_i calculated in the ST module. For each one of the three states of TW2, open, closing and closed, we constructed a table, Table 2.3, Table 2.4 and Table 2.5, respectively, that describes the trend to be used at time t_i given the actual trend and the trend used at time t_{i-1} .

Once TW2 is open, it remains open for a slop time $\Delta t(\text{TW2})$. If another p or w variable reaches ϵ_s during $\Delta t(\text{TW2})$, then its trend is set to (↑) or (↓) as long as the variable actual trend (calculated in the ST module) at t_i is (↑) or (↓) and the trend used at t_{i-1} is either (↑) or (↓), or (?) (see Table 2.3). Also, if the variable actual trend at t_i is (-) and the trend used at t_{i-1} is (↑) or (↓), then the trend at t_i is set to (↑) or (↓), respectively. If the variable does not reach ϵ_s during $\Delta t(\text{TW2})$, that is, the variable actual trend at t_i is (-), and the trend used at t_{i-1} is (?), then the trend at t_i is also set to (?). This represents an exact algorithmic treatment of the theory discussed in Chapter 2 of Volume 1.

As discussed in the theory volume, automatic control actions have the potential to cause a discontinuity in the first time-derivative of the variables if the control action (1) tries to reverse the symptoms of the fault; (2) further increases the symptoms of the fault; or (3) causes variables that are originally constant to deviate from their expected values. The first possibility is considered in rows 3 and 4 of Tables 2.3-2.5; if the variable has a nonmonotonic behavior in two consecutive time steps the diagnostic process is halted. This procedure is somewhat more conservative than what the theory volume calls for, as it is tantamount to shutting down the diagnostics on closure of only one TW. Due to the practical difficulty of implementing the second possibility, it was not treated. This

Table 2.3. Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window is Open $t_{\text{open}}(\text{TW2}) < t_i < t_{\text{open}}(\text{TW2}) + \Delta t(\text{TW2})$ for the $\{p, w\}$ Class of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	—	↑
↑	↓	stop
↓	↑	stop
↓	—	↓
↓	↓	↓
?	↑	↑
?	—	?
?	↓	↓

theoretically should be acceptable because it would reinforce the original trend of the deviating variables. The third possibility, which also causes the end of the diagnostics, is considered below in rows seven and nine of Table 2.5 when the TW is already closed and the variable trend changes from (—) to either (↑) or (↓). This is equivalent to treating the break from steady-state as a control action and shutting down diagnosis on control action. Once again this is more conservative than theory and shuts down the diagnostics earlier.

We chose to halt the diagnosis once automatic control actions are detected because, in general, for water systems, by the time control actions are activated the signal trends are already established. Since no additional information is gained (in the form of trends in plant parameters) due to the control actions, there is no need to continue the diagnosis because no further discrimination will be possible. This strategy could, however, prevent the use of slow-changing or time-integrated parameters such as temperatures T and liquid levels ℓ (although this was not our experience in the application of the ES presented in Volume 3). To account for these slow-changing parameters, in a future implementation of the TWS algorithm, we could “freeze” the trend of the fast-changing parameters (p and w)

once control actions are detected, and continue the diagnosis using the current trend of T and ℓ as long as their current trend is found not to be due to control actions.

Table 2.4 presents the trend to be used for the $\{p,w\}$ class of variables when TW2 is closing, that is, $t_i = t_{\text{open}}(\text{TW2}) + \Delta t(\text{TW2})$. The trends to be used at the closing of TW2 are the same as the ones used when TW2 is open (Table 2.3), except for the case when the variable actual trend at t_i is $(-)$ and the trend used at t_{i-1} is $(?)$. In this case, indicated in the next to the last row of Table 2.4, the trend at t_i is set to $(-)$. After TW2 is closed, that is for $t_i > t_{\text{open}}(\text{TW2}) + \Delta t(\text{TW2})$, Table 2.5 is used.

The treatment of the $\{p,\ell\}$ class of variables for separated volumes is very similar to the treatment of $\{p,w\}$ for non-separated volumes. For this class of variables, Table 2.2 should be used for the opening of TW3, $t_i = t_{\text{open}}(\text{TW3})$, Table 2.6 should be used when TW3 is open, $t_{\text{open}}(\text{TW3}) < t_i < t_{\text{open}}(\text{TW3}) + \Delta t(\text{TW3})$, Table 2.7 should be used when TW3 is closing, $t_i = t_{\text{open}}(\text{TW3}) + \Delta t(\text{TW3})$, and Table 2.5 should be used when TW3 has already been closed, $t_i > t_{\text{open}}(\text{TW3}) + \Delta t(\text{TW3})$.

Table 2.4. Previously Used Trend, Actual Trend, and Trend to Be Used at the Closing of the Time Window $t_i = t_{\text{open}}(\text{TW2}) + \Delta t(\text{TW2})$ for the $\{p,w\}$ Class of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	—	↑
↑	↓	stop
↓	↑	stop
↓	—	↓
↓	↓	↓
?	↑	↑
?	—	—
?	↓	↓

Table 2.5. Previously Used Trend, Actual Trend, and Trend to Be Used After the Time Window Has Been Closed $t_i > t_{\text{open}}(\text{TW}) + \Delta t(\text{TW})$ for the $\{p, w\}$ and $\{p, \ell\}$ Classes of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	—	↑
↑	↓	stop
↓	↑	stop
↓	—	↓
↓	↓	↓
—	↑	stop
—	—	—
—	↓	stop

Table 2.6. Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window is Open $t_{\text{open}}(\text{TW3}) < t_i < t_{\text{open}}(\text{TW3}) + \Delta t(\text{TW3})$ for the $\{p, \ell\}$ Class of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	—	?
↑	↓	stop
↓	↑	stop
↓	—	?
↓	↓	↓
?	↑	↑
?	—	?
?	↓	↓

Table 2.7. Previously Used Trend, Actual Trend, and Trend to Be Used at the Closing of the Time Window $t_i = t_{open}(TW3) + \Delta t(TW3)$ for the $\{p, \ell\}$ Class of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	—	—
↑	↓	stop
↓	↑	stop
↓	—	—
↓	↓	↓
?	↑	↑
?	—	—
?	↓	↓

To handle the $\{T\}$ class of variables a different approach is required in order to account for the slow time response associated with thermal inertia. Once opened following the logic presented in Table 2.8, TW4 for T signals remains open (with trends specified through the logic presented in Table 2.9) until the end of the diagnostic session. Except for nonmonotonic behavior that causes the diagnostics to halt, the trend of $\{T\}$ variables inferred in the TWS module are the same as the actual variable trends obtained in the ST module. TW4 never closes to allow us to account for the thermal inertia which normally leads to slower natural T-H feedback. The treatment of the $\{T\}$ class is not applied to catch control system action unless it causes a nonmonotonic trend behavior. By separating w out from the $\{h_i, h_o, w\}$ coupling discussed in the theory section and treating it by procedures for the $\{p, w\}$ coupling, we have once again stopped diagnosis earlier with control action, as the $\{p, w\}$ class of signals responds the fastest. The deletion of a separate treatment for h_i and lumping it together with h_o as a $\{T\}$ class is reasonable, as long as the instrument response time is much smaller than the enthalpy transport time. The issue of h_o (-) discussed in the theory volume is implemented outside the TWS module in the application of particular rules. We wait for one T not normal (/-) trend (not normal (/-) means either increasing or decreasing) before applying the h_o (-) rules, which automatically takes care of the delay time constants between h and w without the need to know the numerical values. This is why we decouple the treatment of h or T from w and approximate the treatment of

$\{h_i, h_o, w\}$ by treating $\{T\}$. A similar treatment is used for $\{\ell\}$ in separated volumes and it is also represented through Tables 2.8 and 2.9. Because we do not have ℓ (-) rules for separated volumes, we may approximate the treatment of $\{\ell, w\}$ with the treatment of $\{\ell\}$ which ignores the delay time constants between ℓ and w . This is why we approximate the treatment of $\{\ell, w\}$.

Table 2.8. Actual Trend and Trend to Be Used at the Opening of the Time Window $t_i = t_{\text{open}}(\text{TW})$ for the $\{T\}$ and $\{\ell\}$ Classes of Variables

Actual Trend at $t_i = t_{\text{open}}$	Trend to be Used at $t_i = t_{\text{open}}$
↑	↑
↓	↓
-	-

Table 2.9. Previously Used Trend, Actual Trend, and Trend to Be Used Once the Time Window Has Been Opened $t_i > t_{\text{open}}(\text{TW})$ for the $\{T\}$ and $\{\ell\}$ Classes of Variables

Trend used at t_{i-1}	Actual trend at t_i	Trend to be used at t_i
↑	↑	↑
↑	-	↑
↑	↓	stop
↓	↑	stop
↓	-	↓
↓	↓	↓
-	↑	↑
-	-	-
-	↓	↓

The algorithm for the TWS module is illustrated in detail in a flowchart in Fig. 2.3 ([2.6] and [2.7]). The algorithm starts after the actual trend of all variables in the entire P&ID has been determined in the ST module and the general TW, TW1, is open. Depending on the state of the TW of each one of the three classes of variables, $\{p,w\}$, $\{T\}$ and $\{\ell\}$, appropriate tables are selected for determining the trend that will be used for each class. Once determined, the same table is used for all variables of a given class in the entire P&ID during that time step. The $\{p,\ell\}$ class of variables for separated volumes is treated differently since each separated volume may have different time constants between p and ℓ and the fact that separated volumes are treated as being decoupled from each other. Each separated volume has its own TW3 and the two variables in a given separated volume are updated through the appropriate tables according to the state of TW3 in each particular volume.

After the appropriate tables for updating each class of variables have been established, a search procedure is initiated where each sensor in the loop is searched one-at-a-time and updated accordingly.

Together, the choices of the slop time Δt for each class of thermal-hydraulically coupled variables and the thresholds ϵ_p and ϵ_s of each variable (which are selected by the user) play important roles in the opening and closing of the TWs. In general, Δt should be long enough for all coupled variables to respond, but short enough that automatic control actions causing trend changes have not been activated. Similarly, ϵ should be small enough to allow all coupled variables to respond, but large enough to delay detection of trend changes caused by automatic control actions and noise in the data. These may not always be possible if the data has too much noise or if the control action response occurs shortly after the opening of the time window.

The values for the slop time Δt for each class of thermal-hydraulically coupled variables can be selected based on instrumentation time response data provided by vendors, through off- or on-line time response tests or through analysis of recorded transient data. The selection of the threshold ϵ should take into account the sensitivity of the instrument and the expected or reference value of the variable. For instance, if the accuracy of a thermocouple instrument is 0.5°F and its reference value is 200°F , then ϵ should not be smaller than 0.25%. For coupled signals, the selection of ϵ should also consider the nonlinear physical relationships between the coupled variables, e.g., $\Delta p = w^2$.

Fig. 2.3. Detailed Flowchart for the Time-Window Selector Module

Fig. 2.3. Detailed Flowchart for the Time Window Selector Module (Cont'd)

Fig. 2.3 Detailed Flowchart for the Time-Window Selector Module (Cont'd)

Fig. 2.3 Detailed Flowchart for the Time-Window Selector Module (Cont'd)

The slop time Δt , primary threshold ϵ_p , and secondary threshold ϵ_s for the four thermal-hydraulically coupled classes of variables, $\{p,w\}$, $\{p,\ell\}$, $\{T\}$ and $\{\ell\}$, used for the Braidwood simulator Chemical and Volume Control System (our test bed presented in Chapter 4) are described in Table 2.10. The reasons for the selection of these constants are based on our strategy of making maximum use of secondary effects while avoiding control action responses. The explanation for the selection of the values for these constants follows.

Table 2.10. Slop Times and Thresholds for the Various Classes of Variables

Variable	Δt (s)	ϵ_p (%)	ϵ_s (%)
$\{p,w\}$	2	5.00	0.40
$\{p,\ell\}$	1 or 10	2.50	0.40
$\{T\}$	--	2.50	0.25
$\{\ell\}$	--	2.50	0.25

For the $\{p,w\}$ class of variables we define the slop time $\Delta t(TW2)$ to be 2 seconds long, and ϵ_p and ϵ_s to be 5.00% and 0.40%, respectively. These choices are made such that $\Delta t(TW2)$ is long enough and ϵ_s is small enough to guarantee that all pertinent p and w trends are used. There is some risk in selecting a "large" $\Delta t(TW2)$. Analysis of the Chemical and Volume Control System (CVCS) transients shows that the control actions are either instantaneous or take, in general, a few seconds (> 2 seconds) to be activated. For the case of instantaneous control action, there is nothing we can do about it since the system can not differentiate these responses from direct responses caused by the fault. For the case of slower control action response, $\Delta t(TW2)=2$ seconds should still be short enough to prevent the system from taking the control action trends into consideration. Since at this stage we do not have noise in the data, a smaller ϵ_s is to our benefit; it helps catch all pertinent p and w trends within the slop time. When noise is added to the data, we may need to re-examine the selection of the secondary threshold.

For the $\{p,\ell\}$ class of variables in separated volumes we define the slop time $\Delta t(TW3)$ to be 1 second long if the p or ℓ variables are responsible for the opening of the general time window $TW1$, and 10 seconds long if they are not. We set ϵ_p and ϵ_s at 2.50% and 0.40%,

respectively. These choices are made such that we can achieve the objectives of detecting instrumentation error before the activation of control actions and still avoid the lack of synchronism between changes in ℓ and p . Instrumentation error activates control actions which, in turn, may cause changes in ℓ or p . Hence, if instrumentation error is not caught before ℓ or p change due to control action, the algorithm will take the two signals as being correlated and will not detect the instrumentation error. To detect instrumentation error we should keep $\Delta t(TW3)$ as small as possible due to the fast control action response associated with the volume control tank in the CVCS. That is the reason we choose $\Delta t(TW3)=1$ second if the opening of the general time-window TW1 is caused by either p or ℓ . To avoid mismatches between these two variables when there is no instrumentation error, we choose a large $\Delta t(TW3)$. $\Delta t(TW3)=10$ seconds seems adequate for this purpose but a larger value could also be selected without affecting the results. A large ϵ_s helps the detection of instrumentation error in the presence of possible control actions and also allows a longer time for the variations in $\{p, w\}$ to be observed in $\{p, \ell\}$. The longer time lag allows us to infer that Q_{mass} is constant in the separated volume.

For the $\{T\}$ class of variables we do not use the slop time Δt and select $\epsilon_p=2.5\%$ and $\epsilon_s=0.25\%$. Because the trends in the T variables are allowed to change from "-" to "/" at any time during the transient once ϵ_s has been reached, i.e., the TW never closes for T , there is no need to select a value for and use Δt . Otherwise, if T is "-" between t_k and $t_k+\Delta t_k$ we would assume that it has an "?" trend during that interval and that it has a "-" trend for $t \geq t_k+\Delta t_k$. We do so because we would not want to assume a "-" trend when in reality it could be a "/" trend. However, if the trend becomes "/" for $t > t_k+\Delta t_k$ we would use it. By doing so, we would defeat the purpose of selecting a value for Δt for T variables; in either case (no Δt or $\Delta t_k \neq 0$) we would use the "/" trend after using a "-" trend. Hence, the proposed approach is to use the face value trend of T at any time, i.e., if ϵ_s has not been reached we infer that T is "-" and if ϵ_s has been reached then T has a "/" trend. This approach is consistent with the first-principles rules of the ES that involve T and the CVCS transients. It will also allow us to take into account the secondary effects with no bearing on possible control action effects. The treatment for the $\{\ell\}$ class of variables is exactly the same as the treatment for $\{T\}$.

2.4.3 Major Routines

In this section we describe the key modules or Prolog predicates (see [2.1] and [2.10]) of the ES. Prolog predicates generally have arguments, and just like in FORTRAN routines,

the arguments are used to pass (or return) information to (or from) the called predicate. To make the code more transparent, we have labeled the arguments passed by a predicate with the "+" (plus) sign in front of the passing argument name and have labeled the arguments returned by a predicate with the "-" (minus) sign in front of the returning argument name. Following are the key predicates of the ES with their corresponding function.

perform_diagnostics(+N,-Status)

- controls the entire diagnostic session where N is the time step and Status is a flag used to halt the diagnostics when it is instantiated to "stop."

get_data (+N)

- reads the input data at every time step from file "input.data", reads the input data from file "input.data2" for the first time step, asserts predicate "component_ss" into file "component_ss.pl" for the first time step and asserts predicate "component" into the current database at every time step. It also sends the information about the measured signals to the graphical interface file "graph_#".

process_comp(+Time,+Loop_name,+Comp_list)

- determines the actual trend of the plant parameters relative to their steady-state values for all components in Comp_list of loop Loop_name at time step Time. This predicate performs the function of the Signal Trend Module discussed in Section 2.4.2.

time_window(+Time,+Misbehave_List1,+Misbehave_List2,-Control_list,-Status)

- determines the trend to be used by the plant parameters associated with the components in the list Misbehave_List2 at time step Time. It returns in Control_list the list of component:sensor_type:sensor_location associated with sensors that have been varying due to system control action, and instantiates Status to "steady_state" or "control_action" if either of the two conditions exist. Misbehave_List1 and Misbehave_List2 are the list of components for which the associated plant parameters have reached the primary threshold and secondary threshold, respectively.

seq_loops(+Time,+Misb_List1,+L_List1,-Seq_list)

- determines the ordered list of loops, Seq_list, that should be searched at time step Time. The ordering of the loops is based on the number of misbehaving signals on the loop. Misb_List1 contains the list of misbehaving components with signals beyond the primary threshold and L_List1 contains the associated loop names for these components.

enthalpy_transport(+Time)

- transports enthalpy, actually temperatures, from the most upstream component of every loop to the subsequent downstream components at time step Time.

const_flow_transport(+Time)

- infers constant flow for the components of the loops of the entire P&ID that have constant flow measurements at time step Time.

search_loops(+Time,-Continue,_,+Seq_list)

- searches over all misbehaving ordered loops of Seq_list at time step Time and tries to apply the first-principles rules (Q rules and CV rules). It may instantiate the flag Continue to "inst_error", if the instrumentation error rules fire; to "error", if there is a conflict in the components or loops; to "htx_break", if a heat exchanger is the faulty component; or to "sep_volume", if a separated volume is the faulty component.

do_diagnostics(+Time,+Loop_name,+Numb,+Junc_name,-Continue)

- applies the rules of modules 00.00 through 110.00 to loop Loop_name at time step Time depending on the value of Numb which is internally determined by the program. Numb is used only as an internal flag for the program. If Numb=1, the rules of module 00.00 through 110.00 are applied, and if Numb=2, then only the rules of module 00.00 through 30.00 are applied. It may instantiate Continue if there is an instrumentation error, inconsistency in the application of the Q rules, heat exchanger break, separated volume malfunction, pipe failure, an increasing or decreasing trend in the Q value of a component has been detected or no unique diagnosis could be performed. Junc_name is the name of the junction during a search through a junction.

instrument_error(+Time,+Loop_name,-Continue)

- determines if there is an instrument error in loop Loop_name at time step Time by applying the rules of module 00.00. If so, Continue is instantiated to "inst_error."

primary_q_rules(+Time,+Loop_name,-Continue)

- applies the rules of modules 10.00, 20.00 and 30.00 at time step Time for loop Loop_name and it may instantiate Continue to "error" if applicable.

htx_chang_temp(+Time,+Loop_name,-Htx_list)

- determines if there are heat exchangers in loop Loop_name at time step Time with varying temperatures in either side. If so, the list Htx_list is instantiated with the heat exchangers.

apply_htx_rules(+Time,+Loop_name,-Continue,_,+Htx_list)

- applies the rules of module 40.00 at time step Time for the heat exchanger in the list Htx_list in loop Loop_name. If any of the rules fire, Continue is instantiated to "htx_break." Otherwise, it applies the rules of module 50.00 when the trend of Q_{eng} in the heat exchanger is unknown, and finally it tries to infer the flow through the heat exchanger by applying one of the rules of module 60.00.

apply_other_rules(+Time,+Loop_name,-Continue)

- applies the CV rules of modules 70.00 and 80.00 at time step Time for the components of loop Loop_name. It also applies the Q rules and CV rules of modules 90.00, 100.00 and 110.00 for separated volumes. If any of the Q rules of module 90.00 is fired, Continue is instantiated to "sep_volume."

seq_loops2(+Seq_list,-Seq_list2)

- re-orders the loops of Seq_list and places the re-ordered loops in list Seq_list2. The ordering is based on the order of Seq_list and the number of times the Q rules of module 20.00 were fired in a loop.

search_loops2(+Time,-Continue,_,+Seq_list2)

- searches over the ordered loops of Seq_list2 at time step Time in order to make a diagnosis. If a diagnosis is made Continue is instantiated to "no" and the loop search is halted. If no diagnosis is made after all loops are searched Continue is instantiated to "yes." Continue can also be instantiated to "inst_error", "pipe" or "error."

stop_or_continue(+Time,+Loop_name,+Junc_name,-Continue)

- determines if there is a Q inconsistency in the loop Loop_name at time step Time. If an inconsistency is found, Continue is instantiated to "error." If an inconsistency is not found and Q is found to be increasing or decreasing, Continue is instantiated to no and no more loops are searched. Junc_name is the name of the junction in a search through a junction.

continue_no(+Time,+Loop_name)

- determines the Q type and trend of the imbalance in loop Loop_name at time step Time. If Q type is mass and there are imbalanced junctions in the loop, then a junction search is

performed in the loops across the junction. If Q type is momentum, then the secondary Q momentum rules of module 120.00 are applied. This predicate also finds the possible faulty component candidates.

junction_search(+Time,+Loop_name,+Junction_list)

- searches through the loops of the junctions included in Junction_list, except for loop Loop_name, at time step Time in order to find faulty components.

continue_no2 (+Time, +Junc_name, +Flow, +B_comps, +Loop_name, +Original_loop, -Flag)

- determines the Q type and trend of the imbalance across the junction Junc_name located in loop Loop_name across the original loop Original_loop at time step Time. B_comps are the possible faulty components upstream/downstream of Junc_name and Flow indicates the direction of the junction flow (to or from the junction). Flag is instantiated to "continue" if the search through the junction should be continued to include a negative logic search.

continue_yes2(+Time,+Junc_name,+Original_loop,+Flow,+B_comps,+Loop_name)

- continues the search across the junction Junc_name located in loop Loop_name across the original loop Original_loop at the time step Time. B_comps are the possible faulty components upstream/downstream of Junc_name and Flow indicates the direction of the junction flow (to or from the junction). The search applies the rules of module 120.00, or the negative Q logic, or the partial logic when Q_{mom} and/or Q_{mass} have not normal (/-) trends.

continue_yes(+Time,-Status,+Seq_list2)

- searches the loops in Seq_list2 at time step Time trying to apply the rules of module 120.00, or determining if there is a problem with a heat exchanger where the secondary side is not represented (in which case Status is instantiated to "stop"), or applying the negative Q logic, or the partial logic when Q_{mom} and/or Q_{mass} have not normal (/-) trends.

2.5 Knowledge Bases

As discussed in Chapter 1, the knowledge base of the ES contains three distinct elements, the Physical Rules Data (PRD), the Component Classification Dictionary (CCD) and the Piping and Instrumentation Database (PID). In this section we describe each one of these

three elements as they are represented in the knowledge base of the ES portion of PRODIAG.

2.5.1 Physical Rules Database

The first-principles rules of the PRD are located in the Prolog file "q_rules2.pl" in the directory ~reifman/weicrada/prolog.1 [2.8]. The file contains three basic types of rules: instrumentation error rules, Q rules, and CV rules. The instrumentation error rules, as their name indicates, are used to detect inconsistencies in the instruments in an attempt to perform signal validation through first principles. The Q rules indicate the type and trend of imbalances in the control volume defined by the location of the components in the T-H system. There are Q rules for generic non-separated volume components, specific non-separated volume components and separated volume components. The CV rules indicate the trend of unmeasured plant parameters for specific components as a function of the Q status of the component and the trend of other plant parameters. There are CV rules for specific non-separated volume components and separated volume components.

The entire set of rules developed for the PRD is described in Section 3.2, PRD Rules, in Volume 1 of this document. Here, we present only the subset of these rules that is actually implemented in the ES. The remaining rules were not incorporated into the PRD because their implementation was not practical. In general, these rules either required unrealistic amounts of instrumentation or provided ambiguous inference.

The rules of the PRD have been grouped in modules according to their type and are presented here in Tables 2.11 through 2.20. The modules and rule numbering scheme presented here (right-hand column of Tables 2.11-2.20) are also used in the Prolog code "prodiag2.0.pl" to allow for cross-correlation of information. These modules are the same ones we referred to in Section 2.4.1 when we described the logic flow of the ES. The nomenclature in the tables is similar to the one used in Section 2.4.2 where Δp indicates pressure drop across a component, p indicates pressure, T indicates temperature, w indicates flow, ℓ indicates level and Q_{mass} , Q_{eng} and Q_{mom} represent imbalance indicators for mass, energy and momentum, respectively. Increasing trends are represented by the symbol " $\uparrow$ ", decreasing trends are represented by the symbol " $\downarrow$ ", constant trends are represented by the symbol "-", and negation is represented by the symbol "/". The subscript "i" indicates inlet, "o" indicates outlet, "hot" indicates the hot side of the heat

exchanger, "cold" indicates the cold side of the heat exchanger, "a" indicates average, "ℓ" indicates level, "up" indicates upstream, and "down" indicates downstream.

The superscripts attached to some of the rule numbers shown in the right-hand columns of Tables 2.11 through 2.20 define the applicability of the rules. Their meanings are as follows:

- a = if there are junctions with Q_{mass}^- , then also infer Q_{mass}^- and Q_{mom}^- in the other loop across the junction until the end or the beginning of the loop (except for closed loops, where it should not be applied). Treat closed loops as open loops starting at the pump.
- b = for open loops the p signal is not necessary. If there are junctions between w measurements then also infer $Q_{\text{mass}}'^-$ and $Q_{\text{mom}}'^-$ in the other loop across the junction until the end or the beginning of the loop. Treat closed loops as open loops starting at the pump.
- c = for open loops the w signal is not necessary. Treat closed loops as open loops starting at the pump.
- d = if there are junctions between w measurements then infer $Q_{\text{mass}}'^-$ and $Q_{\text{mom}}'^-$ in the other loop across the junction until the end or the beginning of the loop. Treat closed loops as open loops starting at the pump.
- e = if all 4 temperature signals are known and the rules do not fire, then infer Q_{mass}^- .
- f = these rules are used on the other side of the heat exchanger if Q_{eng} in this side is unknown together with the following logic:

$$\begin{array}{lcl} Q_{\text{eng,cold}}^- & \Longleftrightarrow & Q_{\text{eng,hot}}^- \\ Q_{\text{eng,cold}}^{\uparrow} & \Longleftrightarrow & Q_{\text{eng,hot}}^{\uparrow} \\ Q_{\text{eng,cold}}^{\downarrow} & \Longleftrightarrow & Q_{\text{eng,hot}}^{\downarrow} \end{array}$$

- g = these CV rules should be used only if the T-H variables are not known, i.e., not measured, and inference should be made at both component input/output.
- h = once an inference about Q_{mass}^- or Q_{eng}^- is made for a separated volume component at time t_i , this inference will be retained for $t > t_i$.

- i = when applying these rules to a closed loop the p_{up} (w_{up}) search should start with the most "upstream" component of the closed loop, i.e., a pump, and the p_{down} (w_{down}) search should start at the very last component before the pump. In no instance should this search be allowed to cross the pump.
- j = this rule should be applied only if the separated volume is the first component of the loop.
- k = this rule should be applied only if the separated volume is the last component of the loop.
- l = if there are junctions between w measurements then also infer Q_{mass} and Q_{mom} in the other loop from the junction until the end or the beginning of the loop. Do not apply this inference if the loop across the junction is a closed loop.

Table 2.11 presents the instrumentation error rules of module 00.00 of the PRD. These rules are used to detect inconsistency in the reading of the instruments as an attempt to perform signal validation through first principles. Module 00.00 contains instrumentation error rules for filters, open valves, heat exchangers (HTX) and separated volumes. In the table we also indicate through the "or" operator the logical order in which the rules should be applied. For instance, if rule 01.10 is activated, then there is no need to try to apply rule 01.20. Actually, if any of the 01.00 rules for filter or open valve apply there is no need to try to apply rules 02.00 and 03.00. When the logical "or" is not present, then all rules should be applied, that is, an "and" logical operator is implicitly assumed. For instance, in Table 2.12 both rules 11.10 and 12.10 should be applied.

Table 2.12 presents the Q rules of module 10.00 used to infer constant (-) trend status in the imbalance indicators for non-separated volumes. The table also identifies the location of the imbalanced control volume for which the rules should be applied. For instance, in rule 11.10 the control volume applies to all components in the loop while in rule 12.10 it applies to the components located between the upstream temperature measurement T_{up} and the downstream temperature measurement T_{down} .

Table 2.13 presents the Q rules of module 20.00 for non-separated volumes for both open and closed T-H loop configurations. The table also identifies the location of the control volumes for which the different inferences should be applied. For example, in rule 21.10,

Q_{mass}^+ applies to the control volume located between the upstream flow measurement w_{up} and the downstream flow measurement w_{down} , Q_{mass}^- applies to the entire T-H loop except for the components located between the two flow measurements w_{up} and w_{down} , and Q_{mom}^- applies to the entire loop. Table 2.14 presents the Q rules of module 30.00 for non-separated volumes for closed loop configurations. Table 2.15 presents the Q rules of modules 40.00 and 50.00 for heat exchangers. Table 2.16 presents the Q_{mom} rules of module 120.00 for open valves, filters, seals and pumps.

Table 2.17 presents the Q rules of module 100.00 used to infer constant (-) trend status in the imbalance indicators of separated volumes. Table 2.18 presents the Q rules of module 90.00 used to infer varying trend status in the imbalance indicators of separated volumes.

Table 2.19 presents the CV rules of modules 60.00, 70.00 and 80.00 that infer the trend of flow w in heat exchangers, pumps, open valves and filters. Table 2.20 presents the CV rules of module 110.00 that infer the trend of temperature T and flow w in separated volumes.

Table 2.11. Instrumentation Error Rules (Module 00.00)

		Instrumentation Error (IE)	00.00
		Filter and Open Valve	01.00
or	or	$\Delta P^{\prime} W^- \Rightarrow IE$	01.10
		$\Delta P^- W^{\prime} \Rightarrow IE$	01.20
	Heat Exchanger (HTX)		02.00
	or	$W_{cold}^- W_{hot}^- T_{i,cold}^- T_{i,hot}^- T_{o,hot}^{\prime} \Rightarrow IE$	02.10
		$W_{cold}^- W_{hot}^- T_{i,cold}^- T_{i,hot}^- T_{o,cold}^{\prime} \Rightarrow IE$	02.20
	Separated Volume		03.00
	or	$P_g^{\prime} \ell^- \Rightarrow IE$	03.10
		$P_g^- \ell^{\prime} \Rightarrow IE$	03.20

Table 2.12. Q Rules with Constant Trend for Non-Separated Volumes (Module 10.00)

Q-Rules Non-Separated Volume 10.00		
	$W^- \Rightarrow Q^-_{\text{mass}} \text{ and } Q^-_{\text{mom}} [\forall \text{ loop}]$	11.10 ^a
	$W^- T_{\text{up}} T_{\text{down}} \Rightarrow Q^-_{\text{eng}} [T_{\text{up}}, T_{\text{down}}]$	12.10
	$W^{\uparrow}_{\text{up}} W^{\uparrow}_{\text{down}} \Rightarrow Q^-_{\text{mass}} [W_{\text{up}}, W_{\text{down}}]$	13.10 ^{i,i}
	$W^{\downarrow}_{\text{up}} W^{\downarrow}_{\text{down}} \Rightarrow Q^-_{\text{mass}} [W_{\text{up}}, W_{\text{down}}]$	13.20 ^{i,i}
	$P^{\uparrow}_{\text{up}} P^{\uparrow}_{\text{down}} \Rightarrow Q^-_{\text{mom}} [P_{\text{up}}, P_{\text{down}}]$	14.10 ⁱ
	$P^{\downarrow}_{\text{up}} P^{\downarrow}_{\text{down}} \Rightarrow Q^-_{\text{mom}} [P_{\text{up}}, P_{\text{down}}]$	14.20 ⁱ

Table 2.13. Q Rules with Varying Trend for Non-Separated Volumes in Open and Closed Loops (Module 20.00)

Primary Q-Rules Non-Separated Volume 20.00		
	$P^{\uparrow} W^{\downarrow}_{\text{up}} W^{\uparrow}_{\text{down}} \Rightarrow Q^{\uparrow}_{\text{mass}}, Q^-_{\text{mass}} \text{ and } Q^-_{\text{mom}}$	21.10 ^b
	$[W_{\text{up}}, W_{\text{down}}] / [W_{\text{up}}, W_{\text{down}}] [\forall \text{ loop}]$	
	$P^{\downarrow} W^{\uparrow}_{\text{up}} W^{\downarrow}_{\text{down}} \Rightarrow Q^{\downarrow}_{\text{mass}}, Q^-_{\text{mass}} \text{ and } Q^-_{\text{mom}}$	21.20 ^b
	$W^{\uparrow} P^{\downarrow}_{\text{up}} P^{\uparrow}_{\text{down}} \Rightarrow Q^{\uparrow}_{\text{mom}}, Q^-_{\text{mom}} \text{ and } Q^-_{\text{mass}}$	22.10 ^c
	$[P_{\text{up}}, P_{\text{down}}] / [P_{\text{up}}, P_{\text{down}}] [\forall \text{ loop}]$	
	$W^{\downarrow} P^{\uparrow}_{\text{up}} P^{\downarrow}_{\text{down}} \Rightarrow Q^{\downarrow}_{\text{mom}}, Q^-_{\text{mom}} \text{ and } Q^-_{\text{mass}}$	22.20 ^c
	$W^- T^{\uparrow}_{\text{down}} T^{\uparrow}_{\text{up}} \Rightarrow Q^{\uparrow}_{\text{eng}} \text{ and } Q^-_{\text{eng}}$	23.10
	$[T_{\text{up}}, T_{\text{down}}] / [T_{\text{up}}, T_{\text{down}}]$	
	$W^- T^{\downarrow}_{\text{down}} T^{\downarrow}_{\text{up}} \Rightarrow Q^{\downarrow}_{\text{eng}} \text{ and } Q^-_{\text{eng}}$	23.20
	$[T_{\text{up}}, T_{\text{down}}] / [T_{\text{up}}, T_{\text{down}}]$	

Table 2.14. Q Rules with Varying Trend for Non-Separated Volumes in Closed Loops
(Module 30.00)

Q-Rules Non-Separated Volume Closed Loops		30.00
or	$W\uparrow_{up} W\downarrow_{down} \Rightarrow Q^{-}_{mass}, Q^{-}_{mass} \text{ and } Q^{-}_{mom}$ $[W_{up}, W_{down}] / [W_{up}, W_{down}] [\forall \text{ loop}]$	31.10 ^d
	$W\downarrow_{up} W\uparrow_{down} \Rightarrow Q^{-}_{mass}, Q^{-}_{mass} \text{ and } Q^{-}_{mom}$	31.20 ^d
or	$P\uparrow_{up} P\downarrow_{down} \Rightarrow Q^{-}_{mom} \text{ and } Q^{-}_{mass} [\forall \text{ loop}]$	32.10
	$P\downarrow_{up} P\uparrow_{down} \Rightarrow Q^{-}_{mom} \text{ and } Q^{-}_{mass} [\forall \text{ loop}]$	32.20

Table 2.15. Q Rules for Heat Exchangers (Modules 40.00 and 50.00)

Secondary HTX Q-Rules		40.00 and 50.00
Q^{-}_{mass} - Rules		40.00 ^e
or	$T/\downarrow_{i,cold} T/\uparrow_{i,hot} T\downarrow_{o,cold} T\uparrow_{o,hot} \Rightarrow Q^{-}_{mass}$	40.10
	$T/\uparrow_{i,cold} T/\downarrow_{i,hot} T\uparrow_{o,cold} T\downarrow_{o,hot} \Rightarrow Q^{-}_{mass}$	40.20
Q_{eng} - Rules		50.00 ^f
or	$W_a^{-} T_o^{-} T_i^{-} \Rightarrow Q^{-}_{eng}$	50.10
	$W_a^{-} T\uparrow_o T_i/\uparrow \Rightarrow Q\uparrow_{eng}$	50.20
	$W_a^{-} T\downarrow_o T_i/\downarrow \Rightarrow Q\downarrow_{eng}$	50.30

Table 2.16. Q Rules for Open Valves, Filters, Seals and Pumps (Module 120.00)

Secondary Q_{mom} - Rules		120.00
Open Valve - Filter - Seal		121.00
$\Delta P \downarrow \quad W \uparrow \quad Q_{mass}^- \Rightarrow Q_{mom}^{\uparrow}$	121.10
	$\Delta P \uparrow \quad W \downarrow \quad Q_{mass}^- \Rightarrow Q_{mom}^{\downarrow}$	121.20
Pump		122.00
$\Delta P \downarrow \quad W \downarrow \quad Q_{mass}^- \Rightarrow Q_{mom}^{\downarrow}$	122.10
	$\Delta P \uparrow \quad W \uparrow \quad Q_{mass}^- \Rightarrow Q_{mom}^{\uparrow}$	122.20

Table 2.17. Q Rules with Constant Trend for Separated Volumes (Module 100.00)

Q-Rules		Separated Volume	100.00 ^h
$W_i^- \quad P_{gas}^- \Rightarrow Q_{mass}^-$		101.10
	$W_o^- \quad P_{gas}^- \Rightarrow Q_{mass}^-$		101.20
	$T_{\ell}^- \quad W_i^{\uparrow} \quad W_o^{\uparrow} \Rightarrow Q_{mass}^-$		101.30
	$T_{\ell}^- \quad W_i^{\downarrow} \quad W_o^{\downarrow} \Rightarrow Q_{mass}^-$		101.40
	$T_{\ell}^- \quad W_i^- \quad T_i^- \quad W_o^- \Rightarrow Q_{eng}^-$		102.10

Table 2.18. Q Rules with Varying Trend for Separated Volumes (Module 90.00)

Q-Rules		Separated Volume	90.00
or	$T_{\ell}/\uparrow$	$\ell\uparrow W_i\downarrow W_o\uparrow \Rightarrow Q\uparrow_{\text{mass}}$	91.10
	$T_{\ell}/\uparrow$	$\ell\uparrow W_i^- W_o\uparrow \Rightarrow Q\uparrow_{\text{mass}}$	91.20
	$T_{\ell}/\uparrow$	$\ell\uparrow W_i\downarrow W_o^- \Rightarrow Q\uparrow_{\text{mass}}$	91.30
	$T_{\ell}/\downarrow$	$\ell\downarrow W_i\uparrow W_o\downarrow \Rightarrow Q\downarrow_{\text{mass}}$	91.40
	$T_{\ell}/\downarrow$	$\ell\downarrow W_i\uparrow W_o^- \Rightarrow Q\downarrow_{\text{mass}}$	91.50
	$T_{\ell}/\downarrow$	$\ell\downarrow W_i^- W_o\downarrow \Rightarrow Q\downarrow_{\text{mass}}$	91.60
	$T_{\ell}\uparrow$	$W_i^- T_i/\uparrow W_o^- \Rightarrow Q\uparrow_{\text{eng}}$	92.10
	$T_{\ell}\downarrow$	$W_i^- T_i/\downarrow W_o^- \Rightarrow Q\downarrow_{\text{eng}}$	92.20

Table 2.19. CV Rules for Specific Components (Modules 60.00, 70.00 and 80.00)

CV - Rules Non-Separated Volume 60, 70 and 80						
HTX CV - Rules						60.00
<u>cold side (q+)</u>						
<div>or</div>	<div>or</div>	Q^-_{mass}	$T^{\downarrow}_o$	$Q^{\downarrow}_{eng}$	$T^{\downarrow}_{in} \Rightarrow W^{\uparrow}_a$	61.10
		Q^-_{mass}	$T^{\uparrow}_o$	$Q^{\uparrow}_{eng}$	$T^{\uparrow}_{in} \Rightarrow W^{\downarrow}_a$	61.20
		Q^-_{mass}	T^-_o	Q^-_{eng}	$T^-_{in} \Rightarrow W^-_a$	61.30
		Q^-_{mass}	T^-_o	$Q^{\uparrow}_{eng}$	$T^-_{in} \Rightarrow W^{\uparrow}_a$	61.40
		Q^-_{mass}	T^-_o	$Q^{\downarrow}_{eng}$	$T^-_{in} \Rightarrow W^{\downarrow}_a$	61.50
<u>hot side (q-)</u>						
<div>or</div>	<div>or</div>	Q^-_{mass}	$T^{\downarrow}_o$	$Q^{\downarrow}_{eng}$	$T^{\downarrow}_{in} \Rightarrow W^{\downarrow}_a$	62.10
		Q^-_{mass}	$T^{\uparrow}_o$	$Q^{\uparrow}_{eng}$	$T^{\uparrow}_{in} \Rightarrow W^{\uparrow}_a$	62.20
		Q^-_{mass}	T^-_o	Q^-_{eng}	$T^-_{in} \Rightarrow W^-_a$	62.30
		Q^-_{mass}	T^-_o	$Q^{\uparrow}_{eng}$	$T^-_{in} \Rightarrow W^{\downarrow}_a$	62.40
		Q^-_{mass}	T^-_o	$Q^{\downarrow}_{eng}$	$T^-_{in} \Rightarrow W^{\uparrow}_a$	62.50
PUMP CV - Rules						70.00 ^g
<div>or</div>	Q^-_{mom}	$\Delta P^{\downarrow}$	$\Rightarrow W^{\uparrow}$			70.10
	Q^-_{mom}	$\Delta P^{\uparrow}$	$\Rightarrow W^{\downarrow}$			70.20
Open Valve - Filter CV - Rules						80.00 ^g
<div>or</div>	Q^-_{mom}	$\Delta P^{\downarrow}$	$\Rightarrow W^{\downarrow}$			80.10
	Q^-_{mom}	$\Delta P^{\uparrow}$	$\Rightarrow W^{\uparrow}$			80.20

Table 2.20. CV Rules for Separated Volumes (Module 110.00)

CV - Rules		Separated Volume				110.00 ^g	
or	Q ⁻ mass	T _ℓ ⁻	ℓ↑	W _i [/] ↑	=>	W _o ↓	111.10 ^j
	Q ⁻ mass	T _ℓ ⁻	ℓ↓	W _i [/] ↓	=>	W _o ↑	111.20 ^j
	Q ⁻ mass	T _ℓ ⁻	ℓ↑	W _o [/] ↓	=>	W _i ↑	111.30 ^k
	Q ⁻ mass	T _ℓ ⁻	ℓ↓	W _o [/] ↑	=>	W _i ↓	111.40 ^k
	Q ⁻ eng	T _ℓ ↑	W _i ⁻	W _o ⁻	=>	T _i ↑	112.10
	Q ⁻ eng	T _ℓ ↓	W _i ⁻	W _o ⁻	=>	T _i ↓	112.20

2.5.2 Component Classification Dictionary

The classification of generic non-separated volume components as a source or sink of mass, momentum or energy used in the CCD database is located in the Prolog file "ccd.pl" in the directory ~reifman/weicrada/prolog.1. Table 2.21 presents the classification of the components as represented in file "ccd.pl". The basis for the classification of the non-separated volume components is presented in Section 3.3, Component Classification Dictionary, of Volume 1 of this document. Separated volume components such as tanks and pressurizers are not explicitly classified in the CCD since the activation of any of the rules of Module 90.00 (presented in Table 2.18) uniquely identifies a separated volume failure. As indicated in Table 2.18, separated volume components are classified a source or sink of mass or energy.

2.5.3 Piping and Instrumentation Database

The PID is located in the Prolog file "pid2.0.pl" in directory ~reifman/weicrada/prolog.1. Table 2.22 presents the six Prolog predicates of "pid2.0.pl" which are used by the ES code "prodiag2.0.pl" as the basis for representing the P&ID of the T-H process to be diagnosed by PRODIAG. These six predicates are modified and extended by "prodiag2.0.pl" during execution as the input file "input.data2" is read and inserted into a new file termed

"component_ss.pl" located in the same directory. We should point out that for simplicity in the representation of the P&ID, pipes between two components are not explicitly represented. They are, however, accounted for during the diagnostics by assuming that between any two components there is always a pipe.

Table 2.21. Classification of Generic Non-Separated Volume Component Types as a Source or Sink of Mass, Momentum or Energy

Imbalance Type and Trend	Generic Component Type
Mass Source	closed valve break junction
Mass Sink	closed valve break junction
Momentum Source	pump open valve seal turbine
Momentum Sink	pump open valve seal filter demineralizer pipe turbine
Energy Source	heat exchanger heater bearing
Energy Sink	heat exchanger heater bearing

Table 2.22. Prolog Predicates Used to Describe the Piping and Instrumentation Diagram of the Thermal-Hydraulic Process to Be Diagnosed

<i>loop</i> (Loop Name, Loop Type, Loop Complexity, Fluid Type, Q Values, Boundary Components, Component List, Component Edges)
<i>component_list</i> (Component List)
<i>component_0</i> (Component Name, Component Loop, Component Type, Fluid State, Volume Type, Surrounding Component, Plant Data, Component Status)
<i>surr_component_0</i> (Surrounding Component, Level)
<i>adjacent</i> (Component X, Loop of X, Component Y, Loop of Y)
<i>component_belong</i> (Component Name, Loop List)

In the following we describe the meaning of each one of the attributes of the six Prolog predicates in Table 2.22. In Prolog (see [2.1] and [2.6]), a predicate may be represented as *predicate_name/#*, where *predicate_name* indicates the name of the predicate and the symbol "#" indicates the number of attributes of the predicate. This predicate nomenclature is used below.

loop/8

Loop Name: Is the name or ID used to characterize each loop, e.g., loop_1, loop_2.

Loop Type: Is the loop type which can be an externally connected system (ecs), internally connected system (ics), secondary system (ss) or "normal" system (ns). For definitions of loop types please see Section 3.1 of Volume 1, ES Taxonomy, of this document.

Loop Complexity: Is the complexity of the loop which can be either closed (closed:2) or open (open:1).

Fluid Type: Is the type of the fluid of the T-H process which can be either liquid or gas.

Q Values: This is a list of Q values indicating the type (mass, energy and momentum) and trend (increasing, decreasing and normal) of the imbalance in the loop. Q_{gas} is also included but it is only used as a slot to set flags during the execution of "prodiag2.0".

Boundary Component: Initially it is set to an empty list, i.e., []. During the execution of "prodiag2.0" this slot is used to indicate the number of times a Q rule with non-constant trend was fired in the loop. This is used to order the loops for the second iteration (search) over the number of misbehaving loops.

Component List: This is an ordered list (based on the flow direction) of all components that form the T-H loop.

Component Edges: This is a list of the pairs of consecutive components of Component List.

component_list/1

Component List: This is a list of all components of the P&ID that appear in all loops.

component_0/8

Component Name: Is the name or ID used to characterize the component.

Component Loop: Is the name of the loop that the component belongs to.

Component Type: Is the generic type of the component, e.g., pump, open valve, junction. This attribute is instantiated in the PID through the input file "input.data2".

Fluid State: Is the physical state of the fluid in the component, e.g., single-phase liquid, two-phase, etc. This attribute is instantiated in the PID through the input file "input.data2".

Volume Type: Is the type of the volume that describes the component, e.g., separated volume, non-separated volume. This attribute is instantiated in the PID through the input file "input.data2".

Surrounding Component: The surrounding component defines the "volume" that surrounds the components, e.g., auxiliary building, containment building, service

building, etc. This attribute is instantiated in the PID through the input file "input.data2", but it not used in the current version of "prodiag2.0".

Plant Data: Is the plant parameter data, e.g., temperature, pressure, level, pressure difference and flow, associated with each component. In addition to the parameter value, this attribute also indicates the trend, primary threshold and secondary threshold of each parameter. This attribute is instantiated in the PID through the input files "input.data2" and "input.data."

Component Status: The component status identifies if the component is a possible faulty component (Status = on) or not (Status = off).

surr_component_0/2

Surrounding Component: The surrounding component defines the "volume" that surrounds other components. Possible surrounding components or plant systems are the auxiliary building, containment building, service building, etc. This information is not used in the current version of "prodiag2.0". This predicate was originally intended to provide a spatial connection between the process components and their corresponding plant location.

Level: Is the level measurements (two) of the corresponding surrounding component. This information is not used in the current version of "prodiag2.0". The information about the plant location of a component (its surrounding component) and its level could be used by the ES to narrow down the diagnosis.

adjacent/4

Component X: Is the name or ID of component X which is thermally connected with component Y. Component X belongs to the "hot" side of the connection.

Loop X: Is the loop name or ID of component X.

Component Y: Is the name or ID of component Y which is thermally connected with component X. Component Y belongs to the "cold" side of the connection.

Loop Y: Is the loop name or ID of component Y.

component_belong/2

Component Name: Is the name or ID of the component.

Loop List: Is the list of all loop names that the corresponding component belongs to.

The "pid2.0.pl" file representing the PID knowledge base of the ES can be automatically generated through a series of queries to the user [2.11]. The Prolog program "pid.in.pl", with the executable termed "pid.in", located in directory ~reifman/weicrada/prolog.1 is used for automatic generation of the PID file by querying the user. The generated Prolog file is automatically termed "pid.pl" which should then be renamed to "pid2.0.pl" for usage by "prodiag2.0".

The "pid.in.pl" Prolog program constructs the PID knowledge base by querying the user for specific information about the P&ID. It lays out the general structure of the P&ID such as the number of loops, loop names, component names, etc., as indicated by the six predicates described in Table 2.22. To expedite the construction of the PID knowledge base, some of the attributes of the predicates in Table 2.22, e.g., component fluid type and state, plant parameter thresholds, are entered through the input file "input.data2" presented in Section 2.2.2, instead of querying the user.

REFERENCES

- [2.1] Quintus Corporation, Palo Alto, California, Release 3, 1991.
- [2.2] J. Reifman, "Description of the Prolog Program that Generates the Knowledge Base for the Representation of the P&ID," Memorandum to T. Y. C. Wei, 11 June 1993.
- [2.3] J. Reifman, "Incorporation of the Time-Window Selector in PRODIAG," Memorandum to T. Y. C. Wei, 26 January 1994.
- [2.4] J. Reifman and T. Y. C. Wei, "PRODIAG - Dynamic Qualitative Analysis for Process Fault Diagnosis," *Proceedings of the 9th Power Plant Dynamics, Control and Testing Symposium*, pp. 40.01-40.15, Knoxville, Tennessee, May 24-26, 1995.

- [2.5] J. Reifman, "Resolution of Component and Loop Q Conflicts," Memorandum to T. Y. C. Wei, 20 July 1993.
- [2.6] J. Reifman, "Algorithm for the Time-Window Selector in PRODIAG 2.0," Memorandum to T. Y. C. Wei, 21 April 1994.
- [2.7] J. Reifman, "Revised Algorithm for the Time-Window Selector in PRODIAG 2.0," Memorandum to T. Y. C. Wei, 10 May 1994.
- [2.8] J. Reifman, "Overall Strategy of PRODIAG Version 2.0 and Rule Modules," Memorandum to T. Y. C. Wei, 5 April 1994.
- [2.9] J. Reifman and T. Y. C. Wei, "Systematic Construction of Qualitative Physics-Based Rules for Process Diagnostic," *Proceedings of the 7th Portuguese Conference on Artificial Intelligence*, pp. 311-322, Funchal, Madeira Island, Portugal, October 3-6, 1995.
- [2.10] Ivan Bratko, "PROLOG Programming for Artificial Intelligence," *Addison-Wesley*, 1986.
- [2.11] J. Reifman, "Description of the Prolog Program that Generates the Knowledge Base for the Representation of the P&ID," Memorandum to T. Y. C. Wei, 11 June 1993.

3.0 THE GRAPHICAL DIAGNOSTICS INTERFACE

As part of the project, a Graphical Diagnostic Interface (GDI) program was developed to provide the user with the diagnostic information inferred by PRODIAG [3.1]. The GDI provides diagnostic information through two windows. The first window provides graphical diagnostic information while the second one provides the same information in a textual form. The GDI is a generic program written in C that allows the user to customize the diagnostic application for the desired process P&ID through a GUI development environment. In this project, the GDI program was run on a UNIX platform under X-windows with SunOS 4.1.3 operating system and a color monitor. However, it can also run on a PC under MS Windows. In this chapter we describe the GDI program and its companion files.

3.1 Overall Architecture of the Graphical Interface

Figure 3.1 shows a schematic of the interface between the ES and the graphics program. It also presents the names of the various files needed to run the GDI. At every diagnostic time step that the ES makes an inference, it automatically generates an output text file termed "graph_#" located in the directory where the ES is running. This file serves as the means of communication between the ES and the GDI and allows for the synchronized presentation of the diagnosis through graphical and textual windows. The symbol "#" represents an integer number starting with the value of "1" for the first diagnostic time step and increasing by one at each subsequent time step. For instance, at the first time step the ES generates "graph_1", at the second diagnostic time step it generates file "graph_2", and so on.

The complete file structure, including the file and directory names, needed to run the GDI program is presented in Fig. 3.2. In the root directory of the user, he/she should have an initialization text file termed ".ptimage". The ".ptimage" file contains information to direct the GDI program to open the display window with the desired "*.pti" file ("crada2.pti" in our case) when the graphics program is initiated [3.1]. This file also sets the path for the directory where the graphics program is going to be run, i.e., ~reifman/weicrada/prolog.1 in our case.

Fig. 3.1. Schematic of the Graphical Diagnostic Interface and Names of the Various Files Necessary to Run the System

Fig. 3.2. Complete File Structure Necessary for Running the Graphical Diagnostics Interface

In the directory where the GDI program is going to be run, that is, ~reifman/weicrada/prolog.1, the user should have the following files: display, ptimage, *.tif and *.pti. The file termed "display" is a script used to run the GDI program. The user can run the GDI program by simply typing "display", or "./display" if there are other programs called "display" in a different directory, after the UNIX prompt. The file termed "ptimage" is the executable C program run by "display". The file called "crada2.tif" contains the raw black-and-white background image displayed by the GDI. It is a compressed bit-map file that can be brought into the system by scanning the image and saving it as a TIFG4 file or by "rasterizing" an electronically generated .DWG file and saving it as a TIFF file. TIFG4 and TIFF (Tagged Image File Format) are two of many formats generally used to compress bit-map files. The *.pti files, crada2.pti and crada3.pti, are text files that hold the description of the objects overlaid onto the raw black-and-white background image of the *.tif file. The *.pti files are automatically created as the result of the customization of the display image described in the next section.

To run the GDI program, it is also necessary to have the configuration files "ptimage.res" and "ptimage.uid" in the directory ~reifman/weicrada/graph. The "ptimage.res" file is a text file used to initialize the background color, foreground color, scrollbar, etc., of the

graphical window and textual window. The "ptimage.uid" file is a binary file that together with the "ptimage" file form the GDI program.

3.2 Graphics Customization

The GDI is a flexible tool that allows for the customization of the graphical display of the process P&ID being diagnosed by PRODIAG. The first step of the customization is to import into the UNIX system the raw black-and-white background image, i.e., the *.tif file described in the previous section, over which the customization will take place. For the Braidwood CVCS used to test bed PRODIAG, the file termed "crada2.tif" served as the raw background image. The image associated with "crada2.tif" is illustrated in Fig. 3.3.

With the background "crada2.tif" file imported into the UNIX system, we can then customize the display image by creating objects that overlay the black-and-white background image. We customize an image by running the GDI program and setting its mode of operation to DESIGN rather than DISPLAY [3.1], and applying and selecting the appropriate pulldown menus, list boxes and dialog boxes in a GUI development environment. Under the DESIGN mode of operation, two types of objects can be created: static and dynamic. Static objects are display objects that remain unchanged throughout the diagnostics, while dynamic objects have attributes that may be updated at every diagnostic time step. There are three types of static objects, strings, lines and boxes, whose attributes, e.g., color, size, are defined during the creation of the object and remain fixed unless the object is edited. Static objects are suitable for adding elements that are missing from the background image. For example, strings could be used for labeling components, while lines and boxes could be used for representing missing pipes and tanks, respectively.

There are four types of dynamic objects: nodes, value, line, and time, where each object type has different attributes that may be updated at every diagnostic time step. Nodes are rectangular boxes used to highlight faulty components. Nodes have only one attribute: color. Value objects are used to display the normalized values of sensor readings. Value objects have two attributes: a numerical value and a color. Dynamic lines are overlaid on the top of the background pipes to indicate blockages or breaks in the piping during diagnostics. Lines have only one attribute: color. Time objects are used to indicate the transient time and have only one attribute: a positive numerical value.

Fig. 3.3. Black and White Scanned Background Image for the Braidwood Chemical and Volume Control System

The result of overlaying static and dynamic objects onto the background image of Fig. 3.3 is illustrated in Figs. 3.4 and 3.5. The image represented in Fig. 3.4 is the image used by PRODIAG as the graphical diagnostics interface for the CVCS and it corresponds to file "crada2.pti". The *.pti files are stored as the user saves the overlaid image at the end of a DESIGN session. The image represented in Fig. 3.5 is the image used to illustrate the locations of the twenty component malfunctions used to test PRODIAG. These locations are marked by four-character labels located inside rectangles, with the labels indicating the type of the malfunction. For instance, CV01 indicates the failure of pump A. For a description of the twenty malfunctions please refer to Table 2.2 of Section 2.2, Transient Database, in Volume 3 of this document. The associated *.pti file is termed "crada3.pti". Reference [3.1] provides additional information on the tailoring of background images for specific applications.

Every dynamic object is identified by a unique name or ID. Dynamic objects are updated by changing the attribute values associated with a given object name or ID at each diagnostic time step. At each diagnostic time step, the ES generates an output text file termed "graph_#" (see Section 2.3.2) which contains the attribute values of the dynamic objects for the corresponding diagnostics. The "graph_#" file is read by the GDI program which updates the graphical and textual diagnostic windows described in the following section.

3.3 Graphical Interface

The GDI program displays the diagnostic information through a textual window and a graphical window. The information in the two windows is redundant. In the textual window the diagnostic is provided through written statements and in the graphical window the information is provided by changing the color of the overlaid dynamic objects to red. The diagnostic information for both windows is interfaced between the ES (which generates the diagnosis) and the GDI (which presents the diagnosis to the user) through the text file termed "graph_#". As discussed in Sections 2.3.2 and 3.1, at the end of the first diagnostic time step the ES automatically generates "graph_1" which is read by GDI and used to update the textual window and the graphical window according to the updated status of the dynamic objects representing the ES diagnosis. Then, the GDI program deletes "graph_1" and starts looking for "graph_2" in order to update the windows for the next time step. After the ES performs the diagnosis at the second time step, it generates

BRAIDWOOD CHEMICAL AND VOLUME CONTROL SYSTEM

Display
Mode

Fig. 3.4. Background Image Overlaid with Static and Dynamic Objects Used for Diagnostics

BRAIDWOOD CHEMICAL AND VOLUME CONTROL SYSTEM TRANSIENT LOCATIONS

Display
Mode

Fig. 3.5. Background Image Overlaid with Static Objects Indicating the Transient Locations

"graph_2" and the GDI uses it to update the windows for the second time step. This sequence is repeated until the diagnostic session is terminated. By making the GDI program look for interface programs with increasing suffixes, i.e., 1, 2, 3, and so on, we guarantee synchronization between the ES diagnosis and the information displayed by the GDI.

Table 3.1 shows the structure of the "graph_#" files. The first column of the table presents the four types of dynamic objects: time, value, node, and line, of the graphical window. The second column of Table 3.1 illustrates each one of the four dynamic objects with a few examples. The last Line object is followed by the word TEXT (used to flag the beginning

Table 3.1. Structure of the "graph_#" Files which Serve as the Means of Communication Between the Expert System and the Graphical Diagnostics Interface

Dynamic Object	Example
Time	5.0
Value	cold_leg3:temp:out yellow 0.9850 cont_sump:level:l1 black 1.0 h2_tank:flow:out black 1.0 . .
Node	cold_leg3 white v_8389a white s_reg_htx red . .
Line	junc_13:rv_sw_htx white rv_sw_htx:junc_16 white unc_16:junc_12 red . .
	TEXT There are no malfunctions in the system EOF

of the information sent to the textual window) followed by the actual text and the word EOF (used to flag to the GDI that it can update the two windows). There is only one entry to the dynamic object "time" which indicates the current transient time, 5.0 seconds, through a positive numerical value. There are as many "value" objects as sensors in the P&ID. "Value" objects are identified by their name or ID, e.g., cold_leg3:temp:out, cont_sump:level:l1, which must coincide with the names used in the customization of the background image stored in the *.pti file, i.e., "crada2.pti" in this case. "Value" objects have two attributes a color, e.g., yellow, black, and a numerical value, e.g., 0.9850, 1.0. "Node" objects are also identified by their name, e.g., cold_leg3, v_8389a, (which must coincide with the names stored in the *.pti file "crada2.pti" and the PID file "pid2.0.pl"), and color is their only attribute. There are as many "node" objects as components in the P&ID. Like "node" objects, "line" objects are defined by their name, e.g., junc_13:rv_sw_htx, (which must coincide with the names stored in the *.pti file), and color is their only attribute. There are as many "line" objects as pipes connecting two components in the P&ID.

In the next chapter we illustrate through a sample case how the user can run the ES and the GDI to emulate an on-line computer-based diagnostic system that would assist nuclear power plant operators during off-normal plant conditions.

REFERENCES

- [3.1] G. Graham and J. Reifman, "The Display PRODIAG Interface and Results for 20 CVCS Transients," Memorandum to T. Y. C. Wei, 4 August 1994.

4.0 SAMPLE PROBLEM

In this chapter we present a sample problem which illustrates the usage of the ES and GDI programs. With the appropriate files for the ES (described in Chapter 2) and the GDI (described in Chapter 3) included in the working directory `~reifman/weicrada/prolog.1`, we can diagnose transients in the CVCS through the ES and display them through the GDI. Here we show the example of the letdown heat exchanger tube leak transient (CV23) with 65% severity, where the file containing the transient, "cv23.65.D", was copied to the working directory from the directory `~reifman/weicrada/norm.braidwood` and renamed "input.data" as described in Section 2.2.1.

Two windows, at the working directory `~reifman/weicrada/prolog.1`, are needed to run the ES/GDI programs. One window is used to activate and run the ES while the second window is used to activate the graphics. The ES window should be run under the computer containing the license for Quintus Prolog [4.1], "foton" in this case, while the GDI window can be run under any computer. To activate and run the ES, the user should type "prodiag2" and press the <return> key in the ES window. To activate the GDI, the user should type `./display` and press the <return> key in the GDI window. Figure 4.1 depicts the two windows and the corresponding names of the programs necessary to run the two programs.

After the `./display` command is invoked under the GDI window, two windows, the CRADA Display and the Diagnostic Information, are automatically generated by the graphics program as illustrated in Fig. 4.2. The "CRADA Display" window is used to provide graphical diagnostics and the "Diagnostic Information" window is used to provide textual diagnostics after the ES is invoked. These two windows can be resized and moved around the computer monitor just like normal windows created under UNIX.

After the "prodiag2" command is invoked under the ES window, the diagnostic system queries the user to type "go." to continue (or start) the diagnostics or "stop." to terminate the diagnostic session as showed in Fig. 4.3. By typing "go." the user causes the ES to read the data from the input file "input.data" for the first time step (see Fig. 4.4), and use these data as the reference steady-state values against which future values will be compared. By typing "go." again, the ES reads the data for the next time step and starts making inference at time step two, i.e., 2.0 seconds. By typing "go." again, the ES reads

the data for the next time step and makes inference at the next time step, i.e., 3.0 seconds, as illustrated in Fig. 4.5. This process can be repeated until the user wants to terminate the diagnostic session, which is performed by typing "stop." followed by a control-d to return to the UNIX prompt.

At every time step that the ES makes an inference, the attributes of the dynamic objects of the graphical window are automatically updated and the new diagnostic information is appended to the textual window. Unlike the information in the textual window, the image in the graphical window can not be re-visited for previous time steps. Figure 4.6 shows the graphical window at 5.0 seconds into the transient where the faulty components candidates of the letdown line from cold leg 3 to the letdown heat exchangers are highlighted (in a color computer monitor they would show in red). Figure 4.7 shows the graphical display at 9.0 seconds where the location and type of the possible faulty components have been reduced to a malfunction between the regenerative heat exchanger and the letdown heat exchanger. Figure 4.8 illustrates the diagnostics information for the first 9.0 seconds of this transient in a textual form. Together, the graphical display and the textual display provide the diagnostic interface from the ES to the user. From the graphical window the user can refresh the GDI window by toggling the "Restart CRADA Input" option under the "CRADA" menu, or exit from the GDI program by selecting "Exit" under the "File" menu.

REFERENCES

- [4.1] Quintus Corporation, Palo Alto, California, Release 3, 1991.

Fig. 4.1. The Two Windows and Corresponding Commands Needed to Run the Expert System and the Graphical Diagnostics Interface

Fig. 4.2. The Graphical Diagnostic Window and Textual Diagnostic Window Automatically Generated by the Graphical Diagnostic Interface Program and Used to Display the Diagnosis

```
foton% pwd
/home/pepper5/reifman/weicrada/prolog.1
foton% prodiag2
% loading file /home/foton1/doa/reifman/prolog.1/prodiag2
% foreign file /home/foton1/sys/local/quintus/generic/qplib3.1.4/librar
y/sun4-4/libpl.so loaded
% prodiag2 loaded in module user, 0.450 sec 238,536 bytes

*****

This program is an Expert System for diagnosing transients in the
in the Chemical and Volume Control System of the Braidwood Plant.

*****

Type "go." to perform diagnostics or type "stop."
to terminate this session: █
```

Fig. 4.3. Expert System Query Used to Control the Diagnostic Session

```
foton% pwd
/home/pepper5/reifman/weicrada/prolog.1
foton% prodiag2
% loading file /home/foton1/doa/reifman/prolog.1/prodiag2
% foreign file /home/foton1/sys/local/quintus/generic/qplib3.1.4/
library/sun4-4/libpl.so loaded
% prodiag2 loaded in module user, 0.467 sec 238,536 bytes

*****

This program is an Expert System for diagnosing transients in the
in the Chemical and Volume Control System of the Braidwood Plant.

*****

Type "go." to perform diagnostics or type "stop."
to terminate this session: go.

The data at 1.0s is used as the reference steady-state value.

% compiling file /home/foton1/doa/reifman/prolog.1/component_ss.pl
% component_ss.pl compiled in module user, 12.500 sec 66,364 bytes
Type "go." to perform diagnostics or type "stop."
to terminate this session: █
```

Fig. 4.4. Expert System Screen After the Diagnostics is Initiated
By Typing "go." at the First Query

```

foton% pwd
/home/pepper5/reifman/weicrada/prolog.1
foton% prodiag2
% loading file /home/foton1/doa/reifman/prolog.1/prodiag2
% foreign file /home/foton1/sys/local/quintus/generic/qplib3.1.4/
library/sun4-4/libpl.so loaded
% prodiag2 loaded in module user, 0.434 sec 238,536 bytes

*****

This program is an Expert System for diagnosing transients in the
in the Chemical and Volume Control System of the Braidwood Plant.

*****

Type "go." to perform diagnostics or type "stop."
to terminate this session: go.

The data at 1.0s is used as the reference steady-state value.

% compiling file /home/foton1/doa/reifman/prolog.1/component_ss.pl
% component_ss.pl compiled in module user, 12.950 sec 66,364 bytes
Type "go." to perform diagnostics or type "stop."
to terminate this session: go.

At 2.0 seconds:
*****

There are no malfunctions in the system

Type "go." to perform diagnostics or type "stop."
to terminate this session: go.

At 3.0 seconds:
*****

There are no malfunctions in the system

Type "go." to perform diagnostics or type "stop."
to terminate this session: go.

```

Fig. 4.5. Expert System Screen After Three Seconds of Transient Diagnostics

BRAIDWOOD CHEMICAL AND VOLUME CONTROL SYSTEM

5.0 s

Fig. 4.6. Graphical Diagnostic Display at 5.0 Seconds Into the Transient

9.0 Σ

Display Mode

Fig. 4.7. Graphical Diagnostic Display at 9.0 Seconds Into the Transient

Fig. 4.8. Textual Diagnostic Display for the First 9.0 Seconds of Transient