DOE/MC/24207--3007

DE91 002087

ISPE: A Knowledge-Based System for Fluidization Studies

1990 Annual Report

S. Reddy

January 1991

Work Performed Under Contract No.: DE-FC21-87MC24207

For
U.S. Department of Energy
Office of Fossil Energy
Morgantown Energy Technology Center
Morgantown, West Virginia

By
West Virginia University
Statistics and Computer Science Department
Morgantown, West Virginia

MASTER

DISCLAIMER

This report 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 any of their employees 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 views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

This report has been reproduced directly from the best available copy.

Available to DOE and DOE contractors from the Office of Scientific and Technical Information, P.O. Box 62, Oak Ridge, TN 37831; prices available from (615)576-8401, FTS 626-8401.

Available to the public from the National Technical Information Service, U.S. Department of Commerce, 5285 Port Royal Rd., Springfield, VA 22161.

DOE/MC/24207-3007
(DE91002087)

ISPE: A Knowledge-Based System for Fluidization Studies

1990 Annual Report

S. Reddy

Work Performed Under Contract No.: DE-FC21-87MC24207

For
U.S. Department of Energy
Office of Fossil Energy
Morgantown Energy Technology Center
P.O. Box 880
Morgantown, West Virginia 26507-0880

By
West Virginia University
Statistics and Computer Science Department
Morgantown, West Virginia 26506

January 1991

Foreword

In the Intelligent Process Simulation Environment Project (IPSE), we applied knowledge based systems techniques to the process simulation domain. This was an interdisciplinary project and required intensive team-work involving a wide variety of *knowledge sources*. The student contributors to the project had background in both engineering/science and computer science. This helped the project tremendously.

Dr. Richard Turton of Department of Chemical Engineering was the *domain expert* throughout the project. He advised on the chemical engineering aspect of the project: the knowledge needed to develop process models, economic and exergy information, heuristics for model analysis, ASPEN simulation and model validation.

Mr. Kevin Williams of DOE/METC provided the initial ASPEN knowledge, systems installation and input data for some of the simulation models.

Shailesh Potnis contributed to the project from its inception, first as a student, then as a research associate. He developed the interactive user interface, IKAM: the knowledge acquisition module, the IPSE knowledge base, cost module and implemented various techniques to increase the efficiency of the IPSE system. He integrated all the subsystems of IPSE and is responsible for the IPSE systems manual and the Users Manual.

Devendra Godbole developed the initial version of the IPSE analyzer, goal extraction facilities and implemented the first prototype for demonstration.

I.N. Balachandran of Chemical Engineering developed various ASPEN flowsheets and input files for ASPEN simulation, implemented THERMOPTER for thermodynamic option selection and helped in developing rules for model analysis.

Sachin Shrotriya, developed the *exergy* analysis module, THERMOCHECK and the knowledge archiving facilities of IPSE.

Contents

1 Executive Summary **1**

2 INTRODUCTION **5**
2.1 Knowledge Engineering 5
2.2 IPSE Operation 6

3 IPSE Architecture **7**
3.1 Knowledge Engineering Environment: LASER 7

4 IPSE Knowledge Base **9**
4.1 Domain Knowledge 10
4.2 Goal Representation 12
4.3 IPSE Rule Base 13
4.3.1 Rule Categories 14
4.3.2 Priority of a Rule 14
4.3.3 Generic Rule Objects 14
4.3.4 Rule groups 16
4.3.5 Local Rules 18

5 The IPSE Analyzer **21**
5.1 IPSE/PLEX Editor 21

6 User Interface **21**
6.1 Graphical Interface 22
6.2 Submodel Identification 22

7 Example Problems **25**
7.1 A Carbon Regeneration Plant 25
7.2 Acrylic Acid Plant 26
7.2.1 A Rule Object for Analyzing Acrylic Acid Plant 28
7.3 Char Gasifier Flowsheet 28

8 Economic Analysis **31**

9 Exergy Analysis **31**

10 THERMOPTER **33**
10.1 The Inference Network 36
10.2 Linking of THERMOPTER with ASPEN 36
10.3 Validation of THERMOPTER 38
10.4 A Session with THERMOPTER 38

11 Thermodynamic Data Checker: THERMOCHECK **40**
11.1 Binary Parameters 41

12 IKAM: IPSE Knowledge Acquisition Module 41

13 IPSE Systems Manual 42

14 IPSE Users Manual 42

15 PROJECT PERSONNEL 45

16 APPENDIX A: IKAM for Rules 46

17 APPENDIX B: IKAM for Submodels 56

18 APPENDIX C: THERMOPTER Session 61

19 APPENDIX D : Exergy Calculations 63

20 APPENDIX E: Script for Explanation 64

List of Figures

1 An Overview of IPSE Architecture 2
2 Architecture of IPSE 8
3a A Generic Submodel 10
3b Distillation Column Submodel 10
4 Representation of A Small Part of The IPSE Knowledge Base 11
5 A Generic Goal Object with Three Members 12
6 An Actual Goal Object for The Acrylic Acid Flowsheet 13
7 A Generalized Rule Object for an Extractor 15
8 A Template of A Global Rule Group with Its Instances 16
9 Char Gasifier Rule Group with its Subrules 17
10 Template of A Local Rule with a Partial List of its Instances 18
11 A subrule of The Char Gasifier Rule Group 19
12 Classification of Rules into Rule Groups 20
13 The Equipment Flowsheet of the Carbon Regeneration Process 23
14 Simulation Flowsheet of The Carbon Regeneration Process 24
15 Acrylic Acid Flowsheet 27
16 An Oxidation Reactor in the Acrylic Acid Flowsheet 28
17 A Generic Rule Object for an Absorber 29
18 Char Gasification Plant 30
19 A Hypothetical Inference Network 35
20 THERMOPTER Inference Network 37
21 The Directory Structure of IPSE 43
22 Available Knowledge for Using IPSE Modules 44
A1 A Rule Group Object Created from The User's Input 52
A2 A Local Rule Object Created from The User's Input 53

List of Tables

1 Exergy Losses in Acrylic Acid Plant 32
2 Exergy Losses Commonly Encounterd in Chemical Process Plant 33
3 THERMOPTER Test Cases 39
4 Option Set SYSOP10 of ASPEN 40
C1 Thermodynamic Option Sets in ASPEN 62
D1 Exergy values for Carbon Regeneration Plant 63

1 Executive Summary

Chemical engineers use mathematical simulators to design, model, optimize and refine various engineering plants/processes. This procedure requires the following steps: 1) preparation of an input data file according to the format required by the target simulator; 2) excecuting the simulation; and 3) analyzing the results of the simulation to determine if all "specified goals" are satisfied. If the goals are not met, the input data file must be modified and the simulation repeated. This multistep process is continued until satisfactory results are obtained. This research was undertaken to develop a knowledge based system, IPSE (Intelligent Process Simulation Environment), that can enhance the productivity of chemical engineers/modelers by serving as an intelligent assistant to perform a variety tasks related to process simulation. ASPEN[1], a widely used simulator by the U.S. Department of Energy (DOE) at Morgantown Energy Technology Center (METC) was selected as the target process simulator in the project.

IPSE, written in the C language, was developed using a number of knowledge-based programming paradigms: object-oriented knowledge representation that uses *inheritance* and *methods*, rule-based inferencing (includes processing and propagation of probabilistic information) and data-driven programming using *demons*. It was implemented using the knowledge based environment LASER[2].
The relationship of IPSE with the user, ASPEN, LASER and the C language is shown in Figure 1.

IPSE consists of the following subsystems:

User Interface - The interface facilitates interaction between the user and the IPSE system using both graphical and textual modes.

Knowledge Base - IPSE knowledge base contains various *types* of knowledge: description of a variety of equipment and ASPEN units, rules for model analysis from parametric as well as thermodynamic view points, economic and exergy information . Knowledge, regardless of the type, is expressed using the *object-property-value* found in most knowledge representation system.

Model Analyzer - This module performs parametric analysis of the flowsheet to make recommendations if the models goals are not satisfied. It is based on the *Rule interpreter* of LASER. The analyzer is also capable of providing an explanation of the solution it proposes. The results of analysis can be archived for quick retrieval at a later time without repeating the analysis.

THERMOPTER - It advises the user on the proper selection of thermodynamic options (ASPEN has thirteen options), a crucial step for successful simulation. THERMOPTER uses the network inferencing facilities provided by the NIE subsystem of LASER to process probabilistic information.

THERMOCHECK - This module checks for the availability of necessary data (in the ASPEN data bank) needed to use a thermodynamic option and provides recommendations on what

[1] Advanced System for Process Engineering.
[2] Registered trademark of Bell Atlantic Software Systems.

An Overview of IPSE Architecture

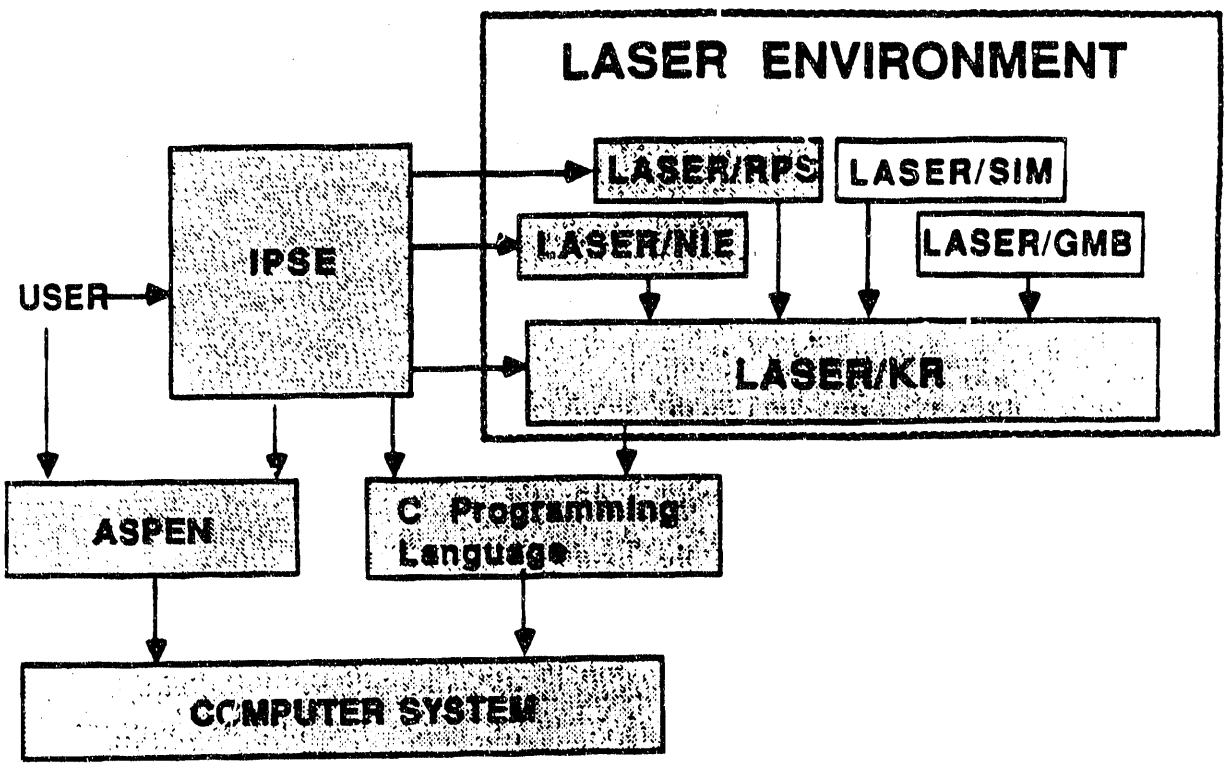


FIGURE 1

to do when the data are missing.

Special purpose modules - An initial version of two additional modules were developed to perform economic and exergy analysis.

Knowledge Acquisition Module - The purpose of this module is to facilitate addition of new knowledge to the system without requiring the user to be a computer programmer.

Since IPSE interfaces with ASPEN, it fetches needed information from the ASPEN Problem Definition File (PDF) to perform a variety of tasks without requiring the user to furnish it.

IPSE knowledge base incorporates the descriptive knowledge of various ASPEN units and "submodels" representing equipment blocks, model goals, cost information and rules for analysis of simulation results. This information (required by the system in "object" form) is organized hierarchically (as classes of objects) for faster access during analysis.

Several flowsheets of varying complexity were selected to demonstrate different capabilitites of IPSE. These include a carbon regeneration facility, an acrylic acid plant and a char gasifier. For each of these flowsheets, a base case ASPEN input data file was prepared but was not optimized to meet the simulation goals. IPSE extracts the model goals from the user and saves them in the knowledge base. After the ASPEN simulation is conducted, it compares the simulation results from the ASPEN PDF file with the model goals to check if all of the model goals are satisfied. If any goal is not satisfied, it triggers the IPSE analyzer which then "travels" through a series of equipment blocks to identify the input parameters to be modified to achieve the user-specified goals. The user can modify the input data file according to the recommendations by IPSE and submit the modified file for resimulation. The process is continued until all the goals are satisfied.

IPSE has the ability to archive results of analysis (identification of the input parameters to be modified). Once the input parameters for modification are identified, the results of analysis can be archived for retrieval at a later time. This avoids repetition of the analysis which can be a time consuming process for large flowsheets. This increases the efficiency of the system. In addition, IPSE analyzer has an "explanation" facility to describe the rationale for its recommendations.

Often an engineer attempting to use ASPEN is not familiar with all the thermodynamic options provided by ASPEN. The selection of appropriate thermodynamic options is very crucial to successful simulation. The THERMOPTER susbsytem of IPSE was developed to provide assistance to the engineer in selecting the proper options in ASPEN. The recommendation by THERMOPTER depends mainly on the operating conditions, type of unit, and nature of components present in a unit. It processes probabilistic information for final recommendation. THERMOPTER is also capable of providing "explanation" of its recommendations.

The THERMOCHECK module of IPSE "finishes" the job initiated by THERMOPTER by checking the ASPEN pure component data bank (PCD), to determine if all the required data are present to use the option recommended by THERMOPTER. In case of missing data, it recommends to the user what to do. THERMOCHECK can also operate on a user-specified choice of thermodynamic option.

The *exergy* analysis module calculates exergy in all the components of the flowsheet. This enables the engineer to focus on areas of high energy losses with a view to further modify the flowsheet for maximizing energy saving.

The *economic* analysis module calculates the Net Present Worth (NPW) using appropriate parameters such as the capital cost, utility cost and revenue over the life of the project.

Knowledge in any knowledge based system like IPSE is never "closed" or complete. It is dynamic and should be updated as needed without any programming effort by the user. To facilitate addition of new knowledge to the IPSE knowledge base, the IKAM (**I**PSE **K**nowledge **A**cquisition Module) subsystem was developed. In an interactive session, IKAM captures knowledge of new equipment and/or new rules of analysis, and converts the information automatically for internal use without requiring the user to write C programs. Since all the knowledge in the IPSE knowledge base is in *object* form, it is also possible to *edit* any object using facilities provided by the knowledge representation module of LASER.

A Systems Manual and Users' Guide document the IPSE system.

Various aspects of the IPSE research were presented at a number of national and international conferences. A research paper on THERMOPTER was the second place award winner in the AIChE student paper contest at the national meeting held in Chicago in November 1990.

2 INTRODUCTION

Chemical engineers depend on process simulation to determine "optimal" plant configurations which are technically feasible and economically viable. Current process simulation technology requires the engineer/designer to follow a lengthy multistep process involving:

1. Develop flowsheet.

2. Translate the plant flowsheet into a simulation flow diagram for the target process simulation system. This step involves preparation of an input file including selection of appropriate thermodynamic options offered by the target simulator.

3. Iteratively execute the model developed in step 2 by altering the values of control variables until specified goals (desired values of output variables) are achieved.

4. Iteratively execute the model determined in step 3 by making structural/topological changes to the flow diagram suggested by exergy and economic analyses until an economically viable and technically feasible design is identified.

This research was undertaken to develop a knowledge based simulation environment IPSE (**I**ntelligent **P**rocess **S**imulation **E**nvironment), that can enhance the productivity of chemical engineers/modelers by serving as an *intelligent* assistant (without requiring the user to be a C programmer) during various phases of process simulation.

IPSE was developed using LASER[3], a tool for developing knowledge based systems. IPSE interfaces with ASPEN[4] and has the capability of extracting relevant information from different ASPEN files: the input file, the PDF (Problem Definition File), and the PCD (Pure Component Data Bank). This minimizes efforts from the users to run the system. Unlike many knowledge based systems, IPSE does not preclude the users from interacting with it at the programmer's level. However, this requires ability to program in the C language in which IPSE and the AI tool LASER were implemented.

IPSE also facilitates enhancement of its knowledge base without requiring the user to program in C. The knowledge base includes both the knowledge about new equipment as well as reasoning knowledge in the form of rules.

2.1 Knowledge Engineering

Building a knowledge based system involves archiving the domain knowledge of the experts. This task/activity of acquiring and transforming the human expertise into a computer-understandable form is often referred to as *knowledge engineering*.

Part of the process simulation knowledge is available from the published documents. However, a major part of this knowledge had to be extracted from the human experts and incorporated into the system incrementally. This is a time consuming process since it involves constant interaction

[3]Registered trademark of Bell Atlantic Software Systems.

[4]Advanced System for Process Engineering.

with the domain experts to ensure consistency and completeness of the information. XCON [5], the rule-based expert system for configuring the computers manufactured by Digital Equipment Corporation required an investment of 50 person-years of effort growing from initial 300 rules to over 4000 rules. For a knowledge based system to be truly useful, (and IPSE is no exception) the role of domain experts cannot be overemphasized; they are to be consulted for model construction, knowledge representation, model analysis, verification of the knowledge base for consistency and completeness, and user interface definition.

2.2 IPSE Operation

The model analysis rules, economic and exergy analyses in IPSE are developed in terms of physical pieces of equipment employed in the plant flowsheet. However, for modeling purposes, more than one ASPEN unit is often necessary to simulate the behavior of each piece of equipment. For example, an inert gas generator may be simulated using two mixers and a *Gibb's Reactor*. It should be noted that this decomposition of a piece of equipment into several ASPEN units is not unique. The behavior of the same equipment may be simulated using other ASPEN units. Since the analysis rules, exergy and cost data should be independent of how one simulates a plant, it is necessary to ascertain the correspondence between the process flowsheet and the simulation flowsheet. For this project it is assumed that a base case ASPEN flowsheet (for which the corresponding input data file exists) is available for simulation. The user is then asked to identify a set of ASPEN units that simulates the performance of each physical piece of equipment in the flowsheet. Once the grouping of all the ASPEN units is done, an equivalent equipment flowsheet is generated which is subsequently used for various types of analysis.
A typical session with IPSE consists of the following activities:

(a) For a given ASPEN flowsheet, group all ASPEN units into a set of predefined "submodels" to generate an equivalent equipment flowsheet.
(b) Identify all the goals for the simulation model.
(c) Submit the input data for ASPEN simulation.
(d) Run IPSE to check if each goal is satisfied. If the goals are not satisfied, the IPSE analyzer employs rule chains to identify appropriate input parameters that must be modified (increase or decrease) to achieve the goals.
(e) Once all the goals are satisfied, perform economic analysis to generate Net Present Worth, utility costs etc.
(f) Compute exergy losses for each piece of equipment in the flowsheet.
(g) Advise the user on appropriate thermodynamic model selection for various simulation units.
(h) Check ASPEN data bank for the availability of required data to use a thermodynamic model and suggest what to do in case of missing data.
(i) Develop new submodels representing new equipment blocks using the IPSE knowledge acquisition module, IKAM.
(j) Develop new rules of analysis at the equipment block level using the rule editing facilities of IKAM.

[5] McDermott, J., " R1: A rule based configurer of Computer Systems", *Artificial Intelligence* 19, 1982.

3 IPSE Architecture

The architecture of IPSE is shown in Figure 2. The IPSE system interfaces with the ASPEN process simulator and consists of the following components which are implemented using the C language.

1. **User Interface.**
   This allows the user to interact with the IPSE system graphically or textually.

2. **Knowledge base.**
   This contains ASPEN knowledge, economic information, exergy calculation procedures, knowledge about various submodels representing equipment blocks, current simulation model and goals, and rules for model analysis as well as advise on thermodynamic options.

3. **IPSE Analyser.**
   This unit performs the analysis of results from ASPEN simulation and provides recommendations to the user for parametric modification.

4. **THERMOPTER.**
   It advises the user on the appropriate selection of thermodynamic options available in ASPEN. The selection is based on the types of components present and the operating conditions of the unit.

5. **THERMOCHECK.**
   This module checks the ASPEN data bank to find out if all the data required to use a certain thermodynamic option are present. In case of missing data, it suggests what steps should be taken to get the data required for the ASPEN thermodynamic model.

6. **IKAM.** This facilitates addition of new knowledge to the IPSE knowledge base.

7. **Special Modules.**
   These include modules for processing the economic and exergy information associated with the flowsheet.

Each of these components are described in the following sections.

3.1 Knowledge Engineering Environment: LASER

The IPSE research combines a number of knowledge-based programming paradigms such as object-oriented knowledge representation with *inheritance*, data-driven programming using *demons* and rule-based programming using heuristics and probabilistic reasoning.

In order to implement various subsystems of IPSE, it was necessary to use an AI tool that supports development of knowledge based systems. An integrated AI environment called LASER (originally developed at WVU Artificial Intelligence Laboratory, now owned and distributed by Bell Atlantic Software Systems) was used to implement IPSE. The LASER system provides a number of programming paradigms and inference engines to facilitate development of a wide variety of knowledge based systems. The following subsystems of LASER were used in the project.

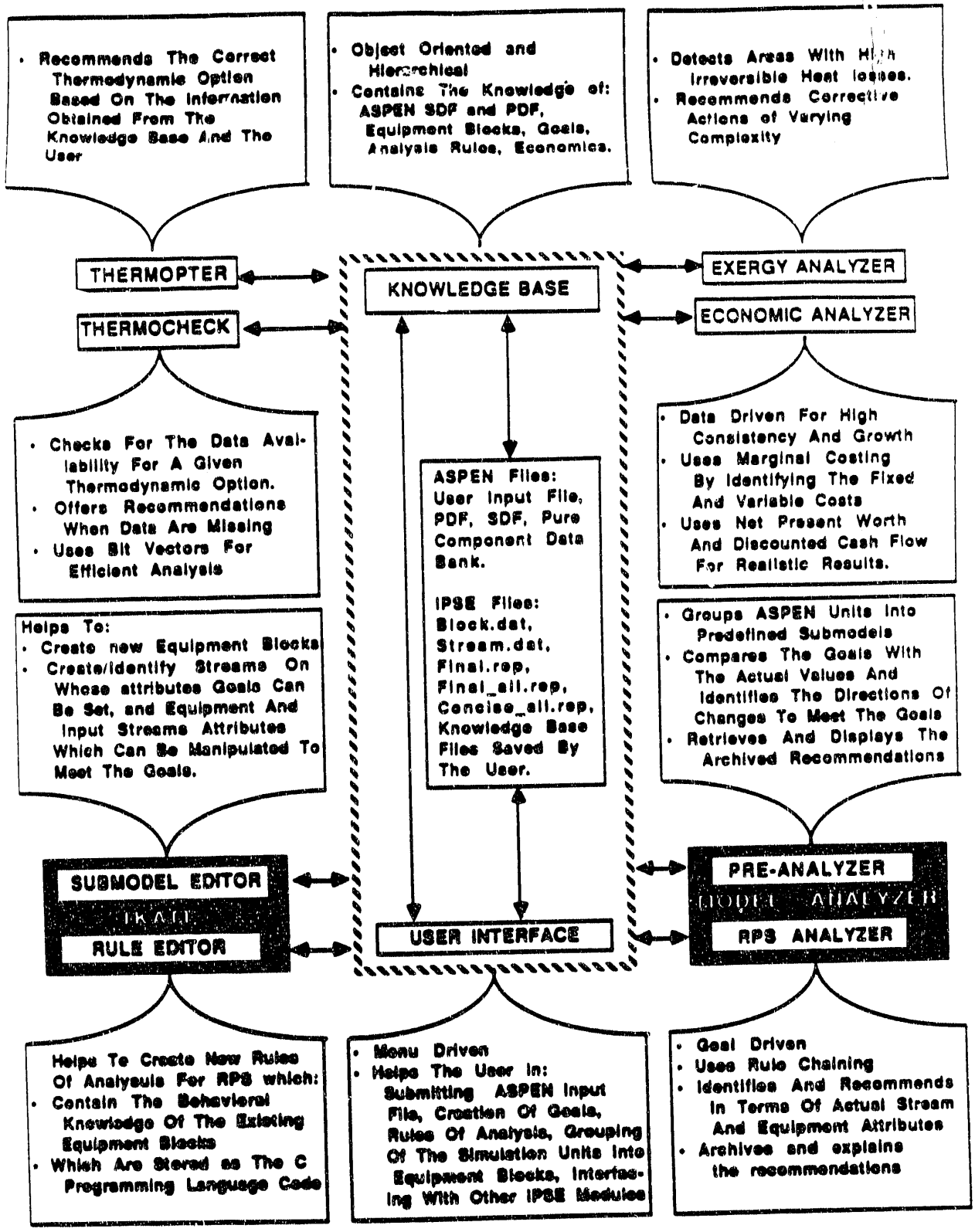


FIGURE 2 The architecture of IPSE

LASER/KR - the object-oriented knowledge representation subsystem to create the IPSE knowledge base in the form *objects* with *properties* and *values*.

LASER/RPS - the forward chaining inference engine to create the IPSE analysis module using rule-based programming.

LASER/NIE - The analysis engine for network inferencing with probabilistic information.

There are two other subsystems (**LASER/GMB** for graphical model building, and **LASER/SIM** for conducting knowledge-based discrete simulation) in the LASER environment which were not used for developing IPSE.

4 IPSE Knowledge Base

For any knowledge based system, the power of the system depends on two factors: the amount of domain-specific knowledge incorporated (properly taxonomized) in the system and its ability to manipulate that knowledge correctly and efficiently. The broad objective of this project was to capture process engineering knowledge and incorporate it into the knowledge base so that IPSE can *intelligently* assist an engineer in problem solving. The knowledge in the IPSE Knowledge base may be classified into the following categories:

1. **Model-independent** declarative knowledge or the domain knowledge that is static. This includes knowledge about various equipment blocks as well as knowledge about a variety of ASPEN blocks and streams.

2. **Current Model** - ASPEN and equivalent equipment flowsheet

3. **Model Goals.**

4. **Basic Economic Information**

5. **Information for the Exergy calculation.**

6. **Problem solving knowledge** that forms the **Rule Base**.

Knowledge in the systems like IPSE is never complete. To facilitate enhancement of the IPSE knowledge base, a knowledge acquisition module (IKAM) was developed to incorporate new knowledge in the system. This new knowledge can be description of a new equipment or new sets of rules for model analysis.

Regardless of the type of the knowledge, it is represented in terms of LASER objects using *object-property-value* format found in many knowledge representation systems. These *objects* form taxonomies (using *isa* hierarchy). Common information is "inherited" *from* the template objects *by* the objects belonging to that class. For example, an actual *distillation column* will inherit all the common features of a "generic" *distillation column* that exists in the knowledge base as a template. Any number of distillation columns may be created by *instantiating* this generic template with actual values associated with the process under consideration. In the following sections we discuss the structure of different classes of knowledge in IPSE.

4.1 Domain Knowledge

The model independent domain knowledge was extracted from the ASPEN System Definition File (SDF) to develop the **generic** IPSE knowledge base[6]. LASER objects are created based on the ASPEN SDF keywords: Primary keywords, Secondary keywords and Tertiary keywords. Knowledge about ASPEN *streams* and a number of ASPEN *blocks* such as *Mixer, Reactors, Heater, Heat Exchanger, Distillation Column, Absorber, Extractor, Separator and Splitter* was incorporated in the IPSE knowledge base.

The knowledge base also contains information about various equipment blocks (*submodels*) such as *absorber, flow splitter, dryer, distillation column, after-burner, extractor, and oxidation reactor.* Each equipment block is a member of the class called *sub_model* shown below.

```
{ sub_model
    inverse_isa # : absorber scrubber dryer activator after_burner
                    gas_generator char_gasifier distillation_column
                    extractor stream_heater flow_mixer oxidation_reactor
                    separator flow_splitter
    inlet * (outlet) <meta_inlet> :
    outlet * (inlet) <meta_outlet> :
    partof * (parts) : main_model
    c_cost <meta_c_cost> :
    cc_factor <meta_cc_factor> :
    specific_info :
    temperature <meta_temperature> :
    total_cost <meta_total_cost> :
   v_c_u <meta_v_c_u> :
}
```

Figure 3a
A Generic Submodel

Rules for model analysis, exergy and economic information were developed at the equipment block level, the level at which engineering analyses are performed.

Following is an example of a generic *distillation column* with a number of common properties that are *inherited* by its *instantiated* members:

```
{ distillation_column
    instances #:
    isa #: sub_model
    bottom_stream <meta_bottom_stream>:
    top_stream <meta_top_stream>:
    feed_stream <meta_feed_stream>:
 }
```

Figure 3b
Distillation Column Submodel

An actual distillation column is created by *instantiating* the above template and filling the inherited properties with actual values that are used in the flowsheet under consideration.

The domain knowledge is organized as a network of generic LASER objects. Figure 4 represents a small part of the generic IPSE knowledge base. Blocks and streams are represented as the **generic primary objects**. Sets of **generic secondary** and **tertiary objects** are also created. Different types of blocks and streams are represented as *instances* of the generic primary objects. Since each

[6]S. Potnis, Problem Report for M.S in Computer Science, West Virginia University, 1988.

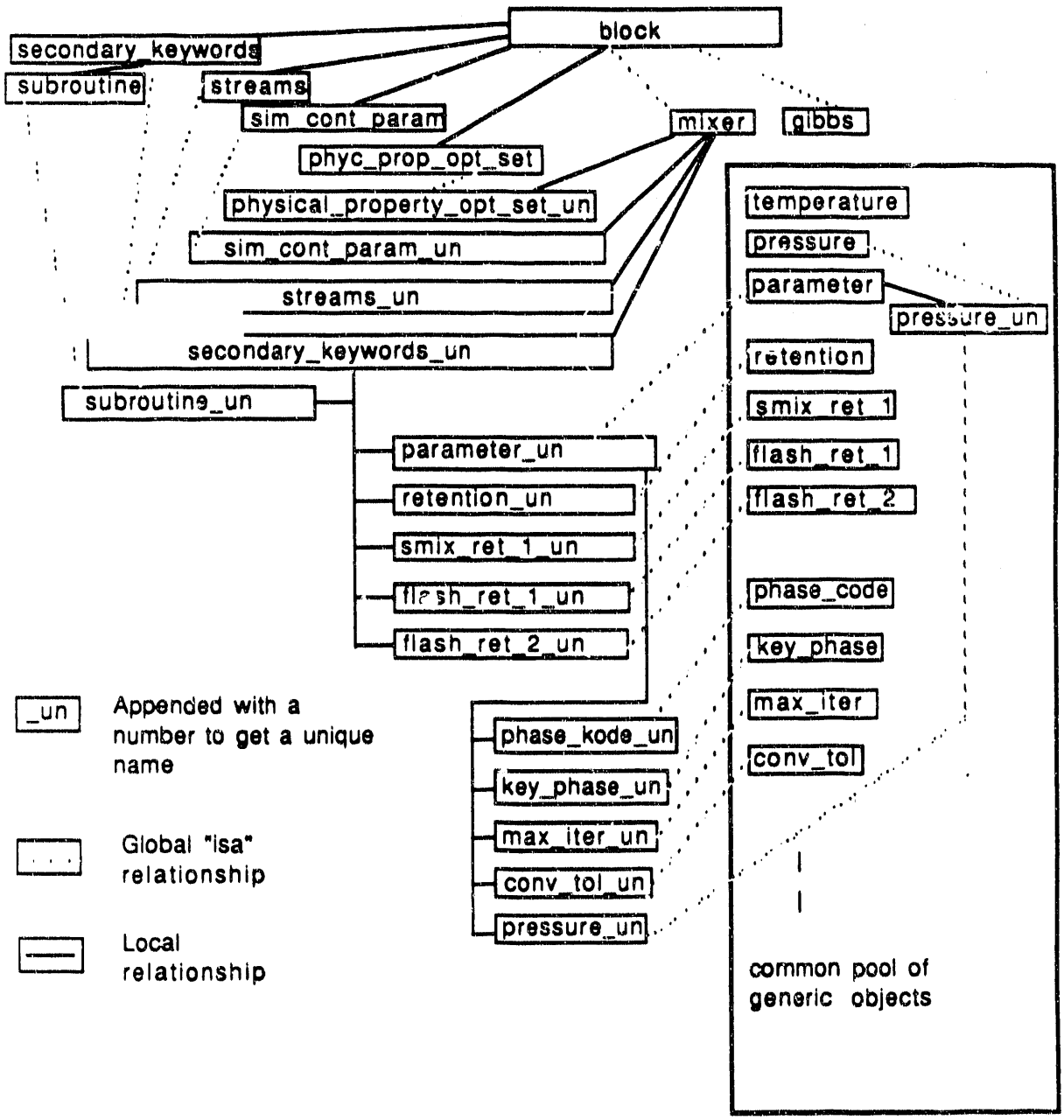


FIGURE 4

Representation of a Small Part of the IPSE Knowledge Base

primary object uses a specific subset of secondary objects, it is related only to this subset through *local relations*. Similarly, a relevant subset of tertiary objects are related to each secondary object. Sets of appropriate *properties* are also created for the primary, secondary and tertiary objects.

4.2 Goal Representation

From the user's input, LASER objects representing goals are created by *instantiating* a *generic* goal object that acts as a template [7]. The instantiated goals inherit all the properties from the generic goal object. The inherited properties are filled with relevant values when a model goal is created. The generic goal object also keeps track of the members that had been created from it. Following is an example of a generic goal object with a predefined set of properties and three instantiated member goals:

```
{ goal
    instances # : goal_1 goal_2 goal_3
    analysis_mode : "GLOBAL"
    analyzed : "no"
    comment :
    direction :
    dummy_object :
    dummy_prop :
    filename :
    fired :
    goal_stream :
    goal_submodel :
    obj_type :
    oper :
    param_obj :
    prescribed :
    satisfied :
    true_object :
    true_prop :
    value :
    visited : "no"
}
```

Figure 5
A Generic Goal Object with Three Members

In the above example, the instantiated goal objects, *goal_1*, *goal_2* and *goal_3* represent the goals for a simulation problem. The template goal keeps track of the number of instantiations created from it in the *instances #* property.

The *goal_1* object created from the generic goal object is shown in Figure 6. The properties are filled with values relevant to the objective of the Acrylic Acid flowsheet discussed later in the report.

[7] D. Godbole, Problem Report for M.S. in Computer Science, West Virginia University, 1988

```
{ goal_1
    instanceof # : goal
    analyzed : "yes"
    comment : "stack's acac (acrylic_acid) mole fraction should be
               less than 1.2e-8"
    direction : "key_component_decrease"
    dummy_object : "stack"
    dummy_prop :
    filename : "stack_acac_decrease"
    fired : "yes"
    obj_type : "stream"
    oper : "<"
    satisfied : "no"
    true_object : "stack"
    true_prop : "key_component" "acac" "acrylic_acid"
    value : "1.2e-8"
    visited : "no"
 }
```

Figure 6
An Actual Goal Object For the Acrylic Acid Flowsheet

The user can set goals on any parameter of any output stream such as temperature, flow rate and composition.

It should be noted that the user is *not required* to input the information in the above format. The IPSE system extracts this information in an interactive session and converts it into "goal objects" for subsequent analysis. The "comment" field was created to provide a description of the goal. Since the goals are also objects, they may be edited directly using the facilities provided by LASER/KR.

After the ASPEN simulation run, the results are checked against each goal. If the goals are not satisfied, IPSE analyzer performs model analysis to provide recommendations to change the appropriate input parameters.

4.3 IPSE Rule Base

In addition to the declarative knowledge about various "objects", the knowledge base also contains problem solving knowledge in terms of rules that describe "what to do" when certain conditions are satisfied. This part of the knowledge base is referred to as the "rule base". There are two types of rules in the IPSE rule base: rules for model analysis and rules for thermodynamic option selection (THERMOPTER rules). While the model analysis rules are interpreted by the LASER/RPS engine, the THERMOPTER rules are processed by the LASER/NIE engine. Only the model analysis rules are described in this section. The rules for THERMOPTER are described in Section 10. Additional rules for model analysis can be developed by the engineers using the IPSE knowledge acquisition module (IKAM) which converts the user-supplied information into rule objects.

Initially, the rules for analysis were developed in terms of actual ASPEN blocks and streams - hard-coded for the simulation problem under consideration. Later the rules were reformulated in terms of generic equipment blocks (submodels) so that their application will be general and not specifically tied to a flowsheet. The model analysis rules describe the relationship between the inputs and outputs of each submodel.

4.3.1 Rule Categories

LASER/RPS has two main categories of rules [8] - simple rules and rule groups. Simple rules specify a set of actions to be executed when a specified set of conditions is satisfied. Rule groups make it possible to organize complex *condition-action* strategies into structured hierarchies. Each category is further subdivided into two classes: global rules and local rules.

LASER/RPS permits two modes of operation: "focussed" mode to solve problems in a hierarchical manner using rule groups and simple rules; and "global" mode, to solve problems in a nonhierarchical manner using only simple global rules.

The rules are written in a standard (IF<condition>THEN<action> format so that they can be interpreted by the inference engine of LASER/RPS. The simple rules consist of <u>at least</u> two properties:

1. LHS - the "IF" part describing a set of conditions. In the "match" phase, the analyzer "interprets" this part of a rule.

2. RHS - the "THEN" part containing the actions to be taken when all the conditions on the left hand side are satisfied and the rule is selected for execution.

The rule groups do not have RHS actions. They have *ENTRY* and *EXIT* actions that are executed by the inference engine. In addition, other properties often used in a rule object are "PRIORITY" (for global rules only) and "COMMENTS". Local and global rule groups also have a property called *CONTROL* that contains a set of conditions similar to the LHS conditions of simple rules. A rule group remains in focus as long as all of its CONTROL conditions are satisfied.

4.3.2 Priority of a Rule

The priority property of a rule has several uses. Selection of a rule from the conflict set may be priority-driven. The priority value may also be used to determine the activity status of a rule. In global mode, the analysis engine evaluates the "if" conditions of each global rule only if the priority of the rule is > 0 (i.e. the rule is active). Thus an active rule may be turned "inactive" by changing the value of the priority property of a rule to 0. This process of "turning off" a rule increases the efficiency of the analysis engine since the analyzer need not interpret the LHS conditions of the rule to determine its eligibility.

4.3.3 Generic Rule Objects

The generic rules of analysis are developed using variables instead of actual streams. In a rule object (Figure 7), the variable names always start with the "?" symbol to signal the rule interpreter for variable processing. From the input/output analysis, the analyzer extracts the names of the actual stream and the appropriate submodel from the current flowsheet, "binds" them to the variables and provides recommendation in terms of the actual streams and components. For a different simulation problem, the stream names and the components may be entirely different.

An example of a rule in a generalized form that operates on an *extractor* is shown in Figure 7.

For a simulation problem (for example, the Acrylic Acid flowsheet, Figure 15), "binding" of variables such as *extract_stream, solvent_stream* and *feed_stream* in the rule object to actual streams

[8] LASER/RPS Users Manual, Bell Atlantic Software Systems

```
{ extractor_rule_1
    instanceof # : LSR_LOCAL_RULE
    SUB_RULE_OF * (SUB_RULES) : extractor_rule_group
    COMMENT : "To increase the flow rate of the extract_stream
               ?extract_stream\n"
               "of the extractor ?extractor, increase the
                 solvent stream \n\'
               ?solvent_stream\n" flow rate and/or increase the feed stream
               ?feed_stream flow rate\n\n"
    LHS : "goal goal_stream = ?extract_stream"
          "?extract_stream extract_stream_of = ?extractor"
          "?extract_stream flow_direction ][ 'analyzed'"
          "?extract_stream flow_direction = 'increase'"
          "?extractor  extract_stream = ?extract_stream"
          "?extractor  solvent_stream = ?solvent_stream"
          "?extractor  feed_stream = ?feed_stream"
    RHS : "lsr_addval ?extract_stream flow_direction 'analyzed'"
          "lsr_addval goal goal_stream ?solvent_stream"
          "lsr_addval goal goal_stream ?feed_stream"
          "lsr_addval ?solvent_stream flow_direction 'increase'"
          "lsr_addval stream_table stream_name ?solvent_stream"
          "lsr_addval stream_table direction 'flow increase' "
          "lsr_addval ?feed_stream flow_direction 'increase'"
          "lsr_addval stream_table stream_name ?feed_stream"
          "lsr_addval stream_table direction 'flow increase' "
          "write '\n To increase the flow rate of the extract stream'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n To increase the flow rate of the extract stream'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n ?extract_stream of the extractor ?extractor, follow the'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n ?extract_stream of the extractor ?extractor,
           follow the'"
           "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n following recommendations:\n'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n following recommendations:\n'"
          "lsr_setval myinit LOG_FILE 'stdout'"
           "write '\n 1) Increase the solvent stream ?solvent_stream flow rate .'"
           "lsr_setval myinit LOG_FILE 'test.'"
           "write '\n 1) Increase the solvent stream ?solvent_stream flow rate.'"
           "lsr_setval myinit LOG_FILE 'stdout'"
           "write '\n 2) Increase the feed stream ?feed_stream flow rate\n\n'"
           "lsr_popval goal goal_stream"
 }
```

Figure 7
A Generalized Rule Object For An Extractor

in the flowsheet takes place and the recommendations for parametric modifications are presented to the user in terms of the actual streams of the flowsheet.

A note about the syntax of the rule object:
In this report several examples of rules in the form of stylized objects are included. They may give the appearance of a complex syntax. However, the user does not have to input the rules in this form. Using the rule editing facilities of the knowledge acquisition module, the engineer can develop rules which are subsequently transformed into "rule objects" as shown in Figure 7 by the system.

The rule objects, like any other objects in the IPSE knowledge base may also be directly constructed/modified using the LASER/KR either at the top level or at the programmer's level. The latter method requires knowledge of the "C" programming language in which LASER and IPSE are implemented.

In order to distinguish the rule objects from other objects in the knowledge base, each rule is implemented as an *instance* of a generic rule object *LSR_LOCAL_RULE*. This improves the efficiency of the engine since the analyzer has to search only through a small "class" of objects for analysis instead of the entire knowledge base.

4.3.4 Rule groups

The rules in the IPSE rule base were organized into "rule groups" for efficient processing by the analysis engine. Each rule group is specific to a submodel representing a piece of equipment.

Each of these groups is created from an existing template in LASER called LSR_GLOBAL_RULE_GROUP shown in Figure 8.

```
{  LSR_GLOBAL_RULE_GROUP
   LHS :
   COMMENT :
   ENTRY_ACTION :
   CONTROL :
   EXIT_ACTION :
   SUB_RULES* (SUB_RULE_OF) :
   PRIORITY : 0
   instances # : splitter_rule_group mixer_rule_group
                 aft_burner_rule_group extractor_rule_group
                 distillation_column_rule_group absorber_rule_group
                 separator_rule_group oxidation_reactor_rule_group
                 steam-heater_rule_group char_gasifier_rule_group
   }
```

Figure 8
Template of a Global Rule Group with its Instances

This template, like other templates in the knowledge base, keeps track of the instances created from it in the property called "instances #".

The char_gasifier_rule_group which is an instance of the above template is shown in Figure 9.

The sub_rule property of the above rule group maintains a list of rules that belong to this class. There are eight rules in the rule base belonging to the char gasifier.

If the conditions in the LHS are satisfied, the rule is placed on the stack and all the sub rules of the rule group become valid candidates. The CONTROL property is similar to LHS but it is tested

```
{ char_gasifier_rule_group
    instanceof # : LSR_GLOBAL_RULE_GROUP
    SUB_RULES * (SUB_RULE_OF) : char_gasifier_rule_1 char_gasifier_rule_2
                                char_gasifier_rule_3 char_gasifier_rule_4
                                char_gasifier_rule_5 char_gasifier_rule_6
                                char_gasifier_rule_7 char_gasifier_rule_8
                                control_rule_2 control_rule_3
    rule_group_of * (rule_group) : char_gasifier
    COMMENT : "This rule group contains the rules and/or rule groups for\n
              the submodel char_gasifier.\n"
              "This rule group is selected when the goal_stream
               ?goal_stream\n"
              "is an outlet of the submodel ?cur_sub which is an instance"
              "of the generic submodel char_gasifier"
    CONTROL : "goal goal_stream = ?goal_stream"
    ENTRY_ACTION : "lsr_setval goal analysis_mode 'FOCUSSED'"
                   "call deactivate_rule_groups"
    EXIT_ACTION : "lsr_setval goal analysis_mode 'GLOBAL'"
                  "call activate_rule_groups"
                 "call explain_ipse_analysis"
    LHS : "goal analysis_mode = 'GLOBAL'"
          "goal goal_stream = ?goal_stream"
         "?goal_stream inlet = ?cur_sub"
         "?cur_sub instanceof = 'char_gasifier'"
    PRIORITY : 20
}
```

Figure 9
Char Gasifier Rule Group with its Subrules

in every cycle. The ENTRY-ACTION is similar to RHS of a simple rule and is executed only once before interpreting the subrules under the rule group.

Explanation of the char_gasifier_rule_group (Figure 9):
The LHS may be interpreted as *"if the analysis mode is global, AND there is still a goal stream to be analyzed. AND the goal stream is an outlet of a piece of equipment, AND the piece of equipment is a char gasifier"*.

If the LHS conditions are satisfied, the ENTRY_ACTION sets analysis mode to "focussed" mode. The CONTROL continually checks if the value of the the goal stream belongs to a char_gasifier. As long as the CONTROL condition of the rule group is satisfied, the sub rules of the rule group will be examined for eligibility by the analyzer.

When the CONTROL conditions are not satisfied, the subrules of the rule group are no longer in focus. The EXIT_ACTION resets the model analyzer to operate in the "global" mode, calls a procedure to activate the rule groups, and invokes the explanation facilities.
The COMMENT field provides a description of the purpose of the rule. It contains variables (starting with the ? symbol) that are replaced with actual values when the rule is used.

4.3.5 Local Rules

All the local rules (these are also subrules belonging to some rule groups) are created from the template of LSR_LOCAL_RULE shown in Figure 10. Since the number of its instances is quite large, only a partial list of its instances is shown in *instances #*.

```
{ LSR_LOCAL_RULE
    LHS :
    COMMENT :
    RHS :
    instances # : gas_generator_rule1 gas_generator_rule2
                  gas_generator_rule3 gas_generator_rule4
                  gas_generator_rule5  gas_generator_rule6
                  gas_generator_rule7 gas_generator_rule8
                  splitter_rule1 splitter_rule2 splitter_rule3
                  splitter_rule4 splitter_rule5 flu_bed_dryer_rule1
                  flu_bed_dryer_rule2 flu_bed_dryer_rule3
                  flu_bed_dryer_rule4 flu_bed_activator_rule1
                  flu_bed_activator_rule2 flu_bed_activator_rule3
                  flu_bed_activator_rule4 mixer_rule1
                  mixer_rule2 mixer_rule3 mixer_rule4 aft_burner_rule1
                  aft_burner_rule2 aft_burner_rule3 aft_burner_rule4
                  aft_burner_rule5 aft_burner_rule6 aft_burner_rule7
                  - - - - - - - - - - - - - - - - -
}
```

Figure 10
Template of the Local Rule with a partial list of its Instances

The LSR_LOCAL_RULE is similar to a global rule but it has no priority and it cannot be in focus by itself. Each local rule must be a sub_rule of some other global rule group or a local rule group which is in focus.

The char_gasifier_rule_1 (Figure 11) is both a local rule (created as an instance of the template in Figure 10) and a sub_rule (of the global rule group called char_gasifier_rule_group).

```
{ char_gasifier_rule_1
    instanceof # : LSR_LOCAL_RULE
    SUB_RULE_OF * (SUB_RULES) : char_gasifier_rule_group
    COMMENT : "To increase the carbon monoxide, ?key_component content\n"
              "of the outlet stream, ?outlet of the char_gasifier
               ?char_gasifier\n"
              "increase the temperature of the ?char_gasifier and\n"
              "decrease the flow rate of steam in the inlet stream ?inlet"
    LHS : "goal goal_stream = ?outlet"
          "?outlet key_component = ?key_component"
          "?outlet tested_key_component ][ ?key_component"
          "?outlet composition_direction = 'increase'"
          "?outlet generic_key_component = 'carbon_monoxide'"
          "?outlet inlet  = ?char_gasifier"
          "?char_gasifier inlet [] ?inlet"
    RHS : "lsr_setval ?char_gasifier temperature 'increase'"
          "lsr_addval ?inlet composition_direction 'decrease'"
          "lsr_addval goal goal_stream ?inlet"
          "lsr_addval ?inlet generic_key_component 'water'"
          "call set_component_id {?retval ?inlet}"
          "write '\n To increase the ?key_component of the outlet stream'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n To increase the ?key_component of the outlet stream'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n ?outlet of the char_gasifier ?char_gasifier,
           follow one or more of'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n ?outlet of the char_gasifier ?char_gasifier,
           follow one or more of'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n  the following recommendations:\n\n'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n  the following recommendations:\n\n'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n 1) Decrease the inlet stream ?inlet ?retval content\n'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n 1) Decrease the inlet stream ?inlet ?retval content\n'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "write '\n 2) Increase the temperature of the
           ?char_gasifier.\n\n'"
          "lsr_setval myinit LOG_FILE 'test.'"
          "write '\n 2) Increase the temperature of the ?char_gasifier.\n\n'"
          "lsr_setval myinit LOG_FILE 'stdout'"
          "lsr_addval ?outlet tested_key_component ?key_component"
          "lsr_popval ?outlet key_component"
          "lsr_popval ?outlet composition_direction"
          "lsr_popval ?outlet generic_key_component"
          "lsr_popval goal goal_stream"
}
```

Figure 11
A Subrule of the Char Gasifier Rule Group

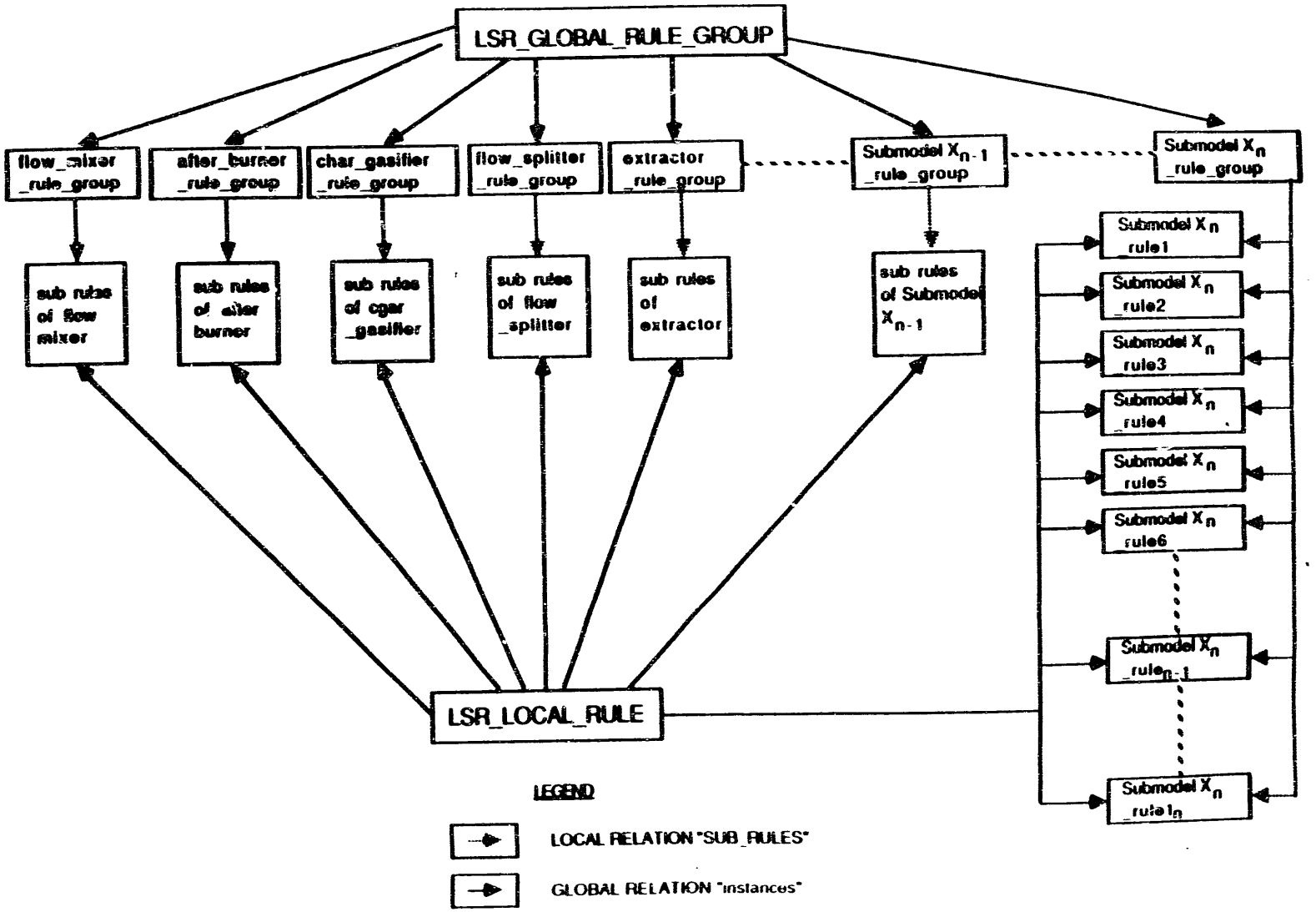


Figure 12

Classification of Rules into Rule Groups

This rule may be interpreted as *"To increase the carbon monoxide content of the outlet stream, decrease the flow rate of the steam, or, increase the temperature of the inlet stream"*

Figure 12 represents the classification of the rules into rule groups and relationship to other rules.

5 The IPSE Analyzer

The model analyzer in IPSE was implemented using [9] the LASER/RPS subsystem. It is assumed that an ASPEN flowsheet (and the corresponding input file) exists for a chemical plant. The user-defined goals for the flowsheet are extracted from the engineer interactively. These goals are saved in the IPSE knowledge base in "object" form for future analysis. In addition to the model goals, the user is also asked to identify the key input streams that are allowed to be modified in the flowsheet. After an ASPEN simulation run, the IPSE analyzer extracts the results of simulation from the ASPEN PDF and compares each goal set by the user to check if it is satisfied. If not, it uses chains of rules to trace through the submodels to identify the key input parameters that should be manipulated (also the direction of modification such as decrease the parameter or increase) to achieve the goal. Intermediate streams in and out of a unit are not considered to be "modifiable", and searching is continued until a key stream is encountered. The process of analysis is continued for each goal that is not satisfied.

The analyzer works in cycles with each cycle consisting of three phases: *Match, Select, Fire.* In the *match* phase, the "IF" conditions are checked and the "matched" rules are stored in the *"conflict set"*. In the "select" phase, one rule from the conflict set is chosen depending on the selection criterion. In the "fire" phase, the actions specified in the selected rule are taken. The cycle stops when there are no rules in the conflict set, or the action part of a fired rule calls explicitly for *"Halt"*.

5.1 IPSE/PLEX Editor

In order to perform analysis of ASPEN simulation, the IPSE analyzer needs to access the simulation results from the ASPEN PDF file. Users of ASPEN have an option of modifying the ASPEN PDF file directly by using an editing program (PLEX) developed at METC. An IPSE version similar to the PLEX editing program was developed using a *subset* of the original PLEX editor (we will call it IPSE/PLEX). The IPSE/PLEX extracts *only* the relevant information that is crucial for analysis and writes all the information to another file. A separate set of routines was developed to convert this information into LASER objects to be used by the IPSE analyzer.

The IPSE/PLEX program also facilitates input modifications suggested by the IPSE analyzer. These modifications are saved in a separate file that may be used for resimulation. The original PDF file is also saved so that the user still has the option of modifying it himself using the METC PLEX editor.

6 User Interface

The user interface allows the user to interact with the IPSE system. Once the system is invoked, various options are presented to the user:

[9] D. Godbole, Problem report for M.S. in Computer Science, West Virginia University, 1988.

specify the model goals/constraints, peruse/alter an already created model, submit the input file for ASPEN simulation, run the IPSE analyzer for parametric modification, perform exergy and economic analysis, invoke THERMOPTER for a thermodynamic model selection, invoke THERMOCHECK for checking the availability of required data in the ASPEN data bank, and invoke IKAM to enhance the IPSE knowledge base.

In order to solve a problem using ASPEN, the user prepares a base case input file (for which the model parameters are yet to be optimized) and conducts a simulation. The ASPEN PDF file contains all the information regarding this model. At this point the IPSE system can be used to capture the model goals from the user and perform other tasks related to simulation.

6.1 Graphical Interface

A graphical interface was developed for grouping ASPEN units into equipment submodels. A menu is displayed in a graphical form from which the user may select his option using a mouse or any other device-specific input device. There are informative messages at the top and the bottom of the menu which guide the users to make a proper selection. This graphical menu is developed using VAX-GKS run-time library and GKS 'C' binding developed in the Artificial Intelligence Laboratory of Department of Computer Science at West Virginia University .

For submodel creation in the graphic mode, a set of routines from the DRAW package (developed at METC) are employed to draw the flowsheet and to identify the ASPEN units to be included in a submodel.

6.2 Submodel Identification

Normally, an engineer starts from a process flowsheet (Figure 13 of the Carbon Regeneration Plant) and from this, a simulation flowsheet is developed (Figure 14).

The decomposition of Figure 13 into the ASPEN units in Figure 14 is not unique. Since IPSE starts with a base case ASPEN flowsheet as in Figure 14, it is important for the IPSE analyzer to have a knowledge of Figure 13: the initial diagram from which the ASPEN flowsheet was generated. The purpose of extracting this information is two-fold:
(1) to apply the IPSE rules for analysis which are written in terms of input-output of specific equipment and *NOT* for ASPEN units, and (2) to provide cost and exergy information about actual physical equipment involved in the flowsheet.
Clearly, these should not depend on how one simulates the plant operation using ASPEN units.

In the IPSE knowledge base, a set of generic submodels are defined which serve as templates for submodels employed in a process flowsheet. Both graphical and textual routines were developed to interact with the user for submodel identification in an interactive session.

IPSE displays the available submodels in the knowledge base and the user is prompted to identify a group of ASPEN units that collectively form a pre-defined submodel.

The flowsheet in Figure 14 consisting of 16 ASPEN units would coalesce into 6 pre-defined submodels shown in Figure 13. Two mixers (*GG-MX1* and *GG-MX2*) and a reactor (*GG-REAC*) are grouped to form a *gas generator*. The process of grouping is continued until all of the ASPEN units are categorized under some pre-defined submodel. Error checking is performed so that each ASPEN unit belongs to only one submodel to ensure that no ASPEN unit is shared among submodels.

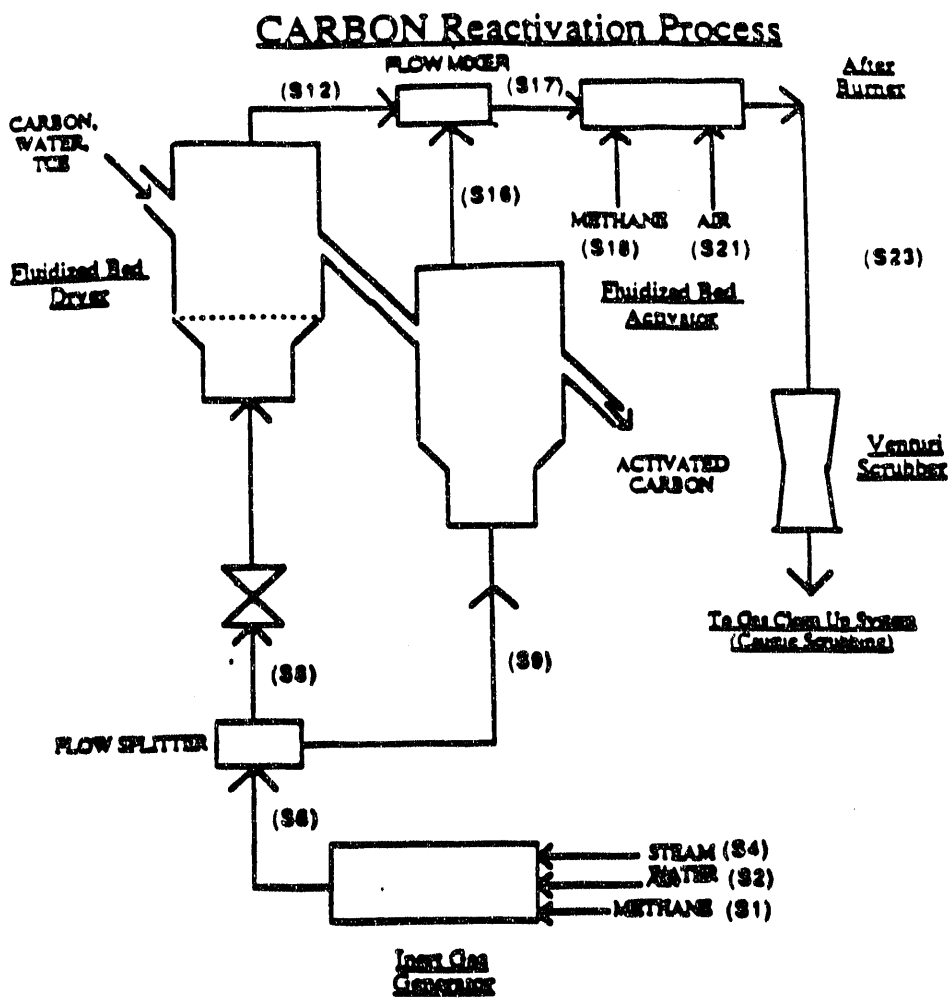


FIGURE 13

The Equipment Flowsheet of the Carbon Regeneration Process

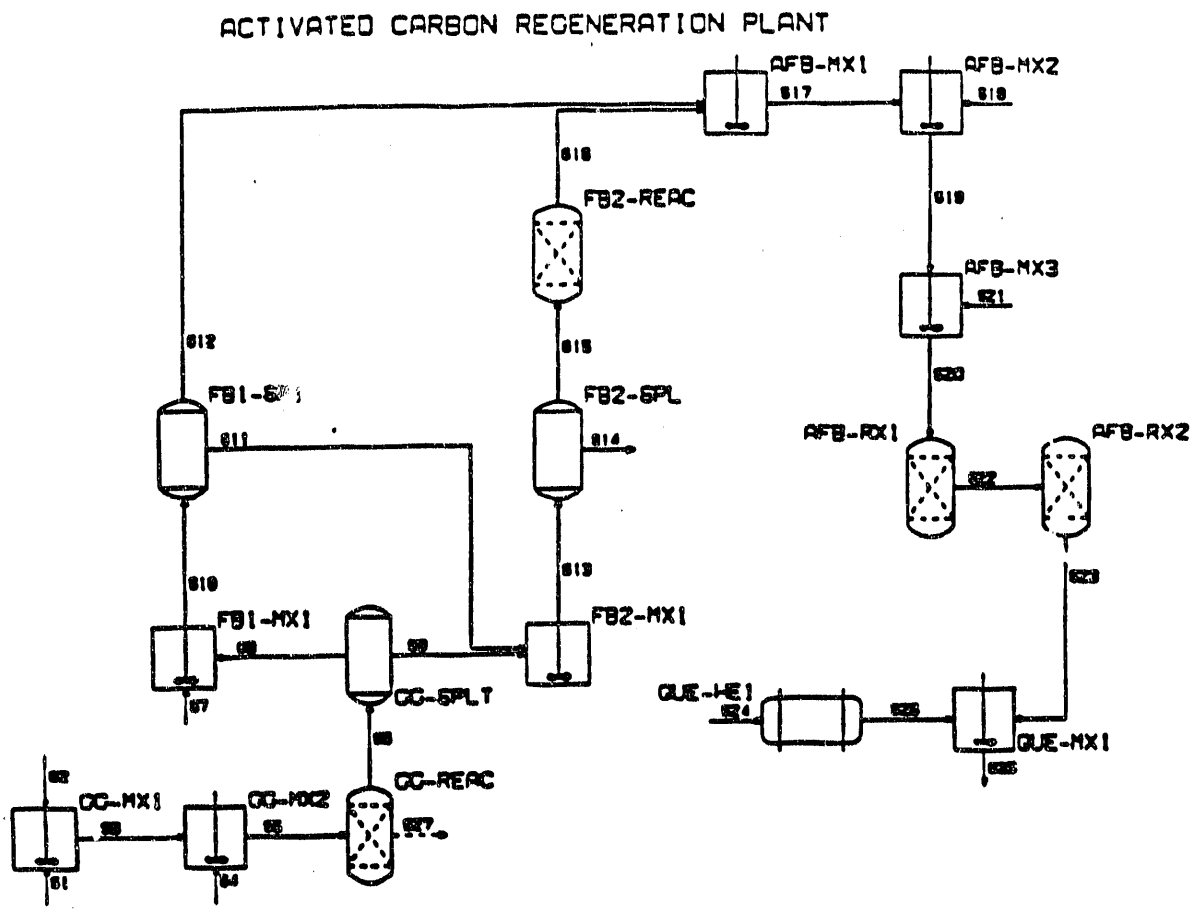


FIGURE 14

Simulation Flowsheet of The Carbon Regeneration Plant

IPSE also ensures that no ASPEN unit in the flowsheet is left out which did not become part of some submodel. In this case IPSE would notify the user the names of ASPEN units which had not been assigned to any submodel.

The users are allowed to create an equipment flowsheet from the equivalent ASPEN flowsheet using only the pre-existing submodels in the knowledge base.

The IKAM module of IPSE, described in Section 12 of this report, enables the engineer to create new submodels if these do not exist in the current IPSE knowledge base.

7 Example Problems

A number of problems of varying complexity were selected for demonstrating the capabilities of IPSE:
(1) A Carbon Regeneration plant,
(2) An Acrylic Acid Plant and
(3) A Char Gasification plant[10].
The purpose of selecting these problems is to expand the scope of the IPSE system by adding a wider variety of information to the IPSE knowledge base. To analyze these problems, it was necessary to add new submodels representing new equipment blocks as well as additional rules of analysis to the knowledge base. A brief description of each of the problems will be given in the following sections with representative rules of analysis.

7.1 A Carbon Regeneration Plant

The example problem that models a carbon regeneration plant employs two fluidized beds to eliminate a carcinogenic chemical (TCE).

The process is shown in Figure 13 . ASPEN flowsheet for the problem is shown in Figure 14.

Process constraints are outlined below.

1. Carbon leaving activator must be reactivated and suitable for resale.
2. TCE content in gas stream leaving the afterburner must be <0.01% of that entering.
3. HCL content in gas discharged to atmosphere must meet EPA regulations.

Typical operating conditions (goals) were :

1. **Inert gas generator**
   Temperature $\simeq$ 2000 F
   %Oxygen $\simeq$ 0.0
2. **Fluidized bed dryer**
   Temperature $\simeq$ 500 F

[10]Char Oil Energy Development, Vol.1, Final Report for Aug 18, 1971 - June 30, 1975 by FMC Corporation: Chemical Research and Development Center, Princeton, NJ.

3. **Fluidized bed activator**

   Temperature $\simeq$ 1600 F

   Carbon gasification $\simeq$ 0.5% of Carbon entering

4. **Afterburner**

   Temperature $\simeq$ 1800 F

   %Oxygen $\simeq$ 3.0% dry basis

5. **Venturi scrubber**

   Stream leaving is a 2-phase mixture at $\simeq$ 14.7 psia.

7.2 Acrylic Acid Plant

The process (equipment) flowsheet is shown in Figure 15. The plant consists of a reactor to oxidize propylene into acetic acid and acrylic acid. The vapors are then sent to a two-stage absorber system where acids are absorbed by water. Acrylic acid is purified using solvent extraction and distillation to remove water and by-product acetic acid.

If a flowsheet has multiple occurrences of the same equipment block, then several instances (with unique names to distinguish them from each other) from the same template can be created. All these instances will have the same set of properties characteristic of the equipment block, but the values attached to the properties may be different. In the example problem of the acrylic acid plant (Figure 15), there are two *absorbers (absorber1* and *absorber2)* and two *distillation columns (distcol* and *stripper*).

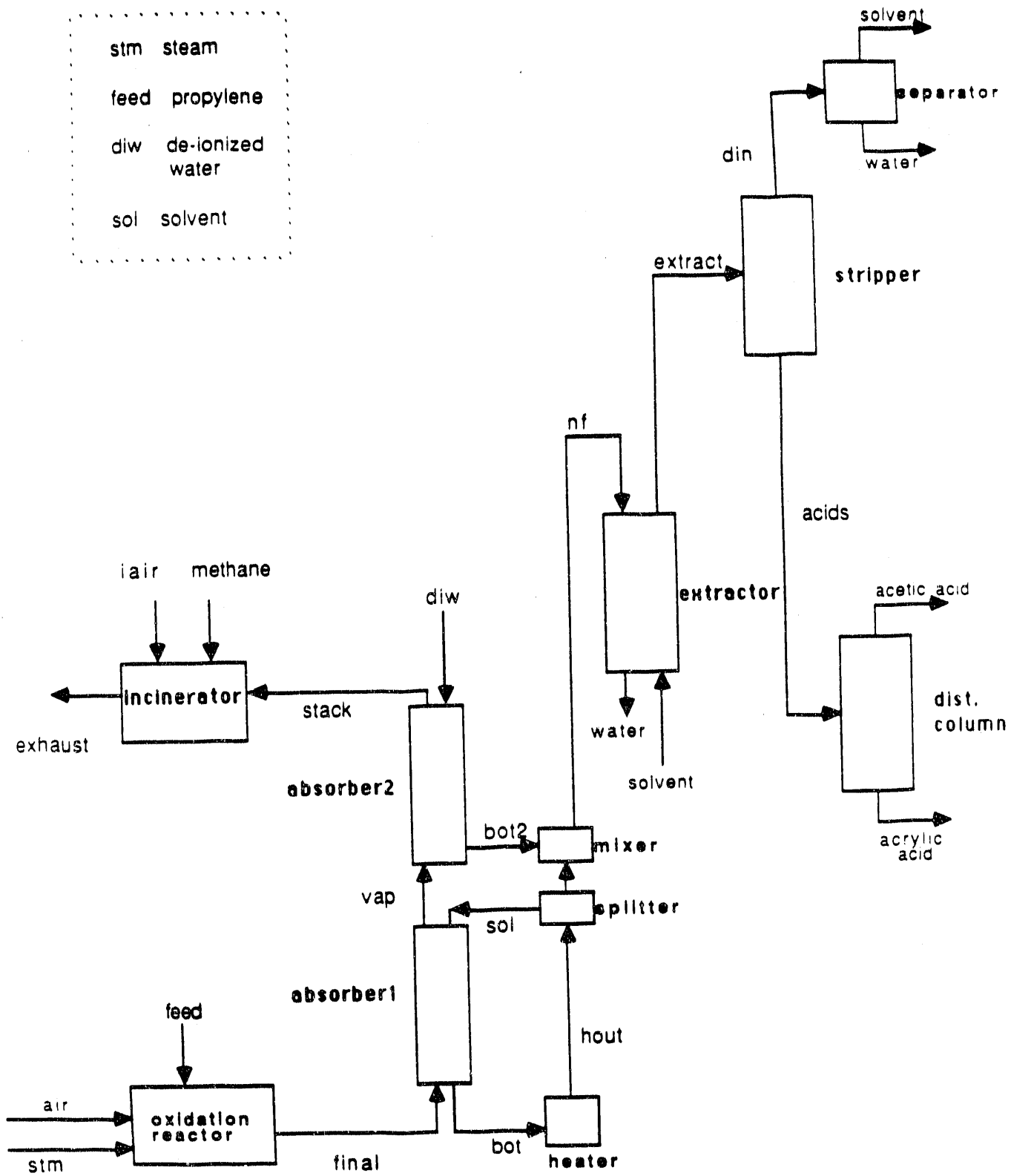


FIGURE 15: Acrylic Acid Flowsheet

Below is an example of an oxidation reactor submodel for the acrylic acid flowsheet.

```
{oxi_reactor
    instanceof # : oxidation reactor
    inlet * (outlet)<meta_inlet> : air feed stm
    outlet * (inlet) <meta_outlet> : final
    partof * (parts) : acry-main_model
    parts * (partof): <meta_parts> : m1 reac
    air stream <meta_air_stream> : "air"
    feed_stream <meta_feed_stream> : "feed"
    steam_stream <meta_steam_stream> : "stm"
}
```

Figure 16

An IPSE Object Representing an Oxidation Reactor in the Acrylic Acid Flowsheet

7.2.1 A Rule Object for Analyzing Acrylic Acid Plant

A generic rule (in object form) for an *absorber* that combines all possible actions to achieve a goal is shown in Figure 17. The purpose of the rule is stated in the COMMENT property of the rule. The recommendations are given to the users in terms of actual streams in the flowsheet after "binding" the variables to the relevant entities.

7.3 Char Gasifier Flowsheet

The flowsheet[11] is shown in Figure 18. The purpose of selecting this problem was two-fold: to apply the generalized IPSE architecture to an energy-related problem, and to enhance the knowledge base for handling additional fossil energy processes. To analyze this problem, it was necessary to add a few submodels representing new equipment blocks and additional rules of analysis to the knowledge base.

In this char oil energy development process (known as COED process), 24 Mtons of char is gasified per day in fluidized bed gasifiers operating at 1530 F. The product gas consists mainly of CO, CO_2, H_2 and H_2O. The fluidized bed gasifier was simulated by a combination of *mixers, splitters* and *reactors* of the type *RSTOIC* and *RGIBBS*. The following reactions were considered.

$$\text{C (CHAR)} + O_2 \rightarrow CO$$

$$\text{CO} + H_2O \rightleftharpoons CO_2 + H_2$$

$$\text{C (CHAR)} + H_2 \rightarrow CH_4$$

$$\text{S (CHAR)} + H_2 \rightarrow H_2S$$

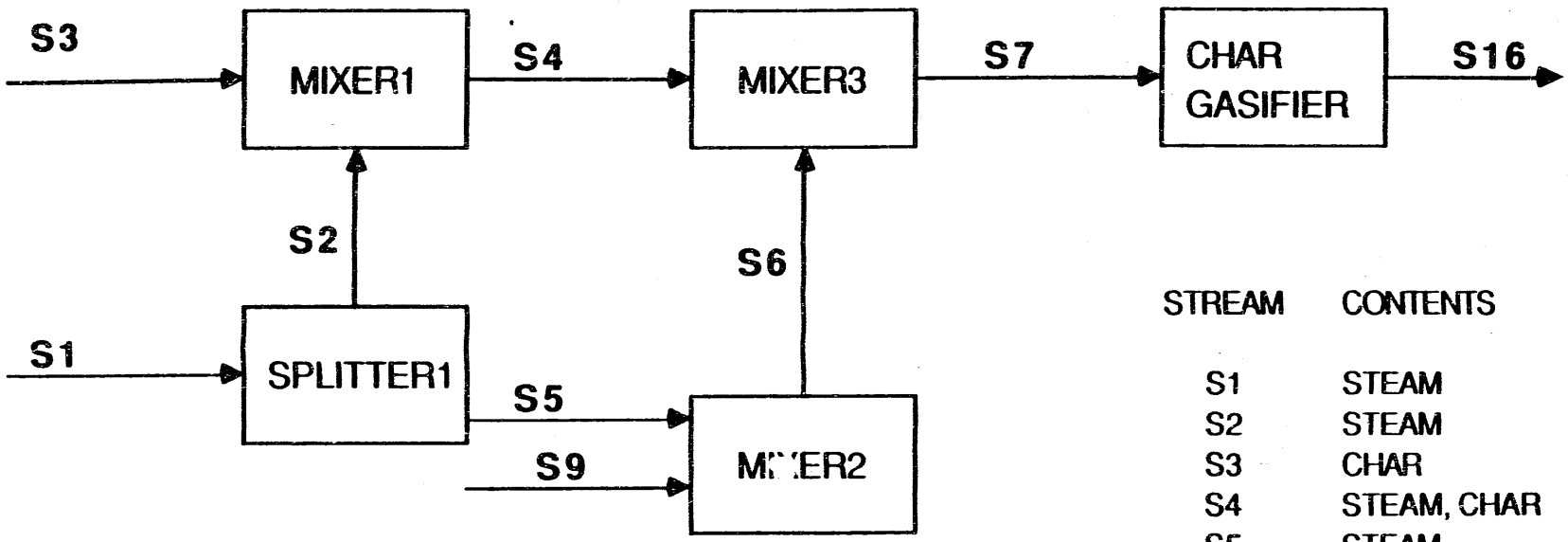
[11]Char Oil Energy Development, Vol.1, Final Report for Aug 18, 1971 - June 30, 1975 by FMC Corporation: Chemical Research and Development Center, Princeton, NJ.

```
{ absorber_rule_2
   instanceof # : LSR_LOCAL_RULE
   SUB_RULE_OF * (SUB_RULES) : absorber_rule_group
   COMMENT : "To decrease the flow rate of the scrubbed stream ?scrubbed_stream\n"
             "of absorber ?absorber, decrease the feed flow rate of feed
              stream ?feed_stream\n"
             "and/or increase the absorbent_stream ?absorbent_stream flow rate"
   LHS : "goal goal_stream = ?scrubbed_stream"
         "?scrubbed_stream scrubbed_stream_of = ?absorber"
         "?scrubbed_stream flow_direction ][ 'analyzed'"
         "?scrubbed_stream flow_direction = 'decrease'"
         "?absorber  feed_stream = ?feed_stream"
         "?absorber  absorbent_stream = ?absorbent_stream"
   RHS : "lsr_addval ?scrubbed_stream flow_direction 'analyzed'"
         "lsr_addval goal goal_stream ?feed_stream"
         "lsr_addval goal goal_stream ?absorbent_stream"
         "lsr_addval ?feed_stream flow_direction 'decrease'"
         "lsr_addval stream_table stream_name ?feed_stream"
         "lsr_addval stream_table direction 'flow decrease' "
         "lsr_addval ?absorbent_stream flow_direction 'increase'"
         "lsr_addval stream_table stream_name ?absorbent_stream"
         "lsr_addval stream_table direction 'flow increase' "
         "write '\n To decrease the flow rate of the scrubbed stream'"
         "lsr_setval myinit LOG_FILE 'test.'"
         "write '\n To decrease the flow rate of the scrubbed stream'"
         "lsr_setval myinit LOG_FILE 'stdout'"
         "write '\n ?scrubbed_stream of the absorber ?absorber,
          follow one or more of'"
         "lsr_setval myinit LOG_FILE 'test.'"
         "write '\n  ?scrubbed_stream of the absorber ?absorber,
          follow one or more of'"
         "lsr_setval myinit LOG_FILE 'stdout'"
         "write '\n  the following recommendations:\n'"
         "lsr_setval myinit LOG_FILE 'test.'"
         "write '\n the following recommendations:\n'"
         "lsr_setval myinit LOG_FILE 'stdout'"
         "write '\n 1) decrease the feed stream ?feed_stream flow rate'"
         "lsr_setval myinit LOG_FILE 'test.'"
         "write '\n 1) decrease the feed stream ?feed_stream flow rate'"
         "lsr_setval myinit LOG_FILE 'stdout'"
         "write '\n 2) increase the absorbent stream ?absorbent_stream flow rate.\n\n'"
         "lsr_setval myinit LOG_FILE 'test.'"
         "write '\n  2) increase the absorbent stream ?absorbent_stream flow rate.\n\n'"
         "lsr_setval myinit LOG_FILE 'stdout'" "lsr_popval goal goal_stream"

 }
```

Figure 17
A Generic Rule Object for an Absorber

CHAR GASIFICATION PLANT



STREAM	CONTENTS
S1	STEAM
S2	STEAM
S3	CHAR
S4	STEAM, CHAR
S5	STEAM
S6	AIR, STEAM
S7	AIR, CHAR, STEAM
S9	AIR
S16	CO, CO2, H20, H2, H2S, CH4, CHAR

FIGURE 18

Only the second reaction was considered to proceed to equilibrium and was simulated using *RGIBBS*. Most of the rules deal with controlling of the composition of the product gas. The control is achieved by changing the temperature of the gasifier or by changing the input flow rate of steam according to Le Chatelier's principle. Since the number of moles does not change during the equilibrium reaction, pressure has no effect on the composition of the products. Two rules (in English) are shown below.

(i) To increase the percentage of CO in the product stream, decrease the flow rate of input steam.

(ii) To decrease the percentage of H_2 in the product stream, increase the temperature of the gasifier.

A rule related to the char gasifier operation and developed using IKAM is shown in Appendix A.

8 Economic Analysis

Since economic analysis is a necessary and crucial step in a plant operation, a basic cost analysis module was developed[12]. This module provides only a rough estimate of the cost and profitability of a given process. Costs are broadly divided into two types: *fixed* or *capital* costs, and *variable* or operating costs. The fixed part is not broken down any further. A future extension may include start-up cost, purchased equipment cost, installed equipment cost, etc.

The variable part depends on the flowsheet parameters which may vary from run to run. In this module, the operating cost was limited to the utilities only, with no allocation for wages or raw material. The utilities are further classified as water, low temperature steam, high temperature steam, electricity and gas. Additional classes of utilities such as coal, special fuels, etc., may be included in a more refined version.

The profitability measure was obtained by computing the Net Present Worth (NPW). It was calculated by taking the difference between the NPW of the total revenue over the estimated life of the project and the sum of the capital cost and the NPW of total utilities. For the Carbon Regeneration plant, the project lifetime was assumed to be 10 years.

Demons (procedures that get automatically invoked when proper conditions are encountered) were *attached* to the *properties* used in the computation of costs. If any parameter contributing to the cost changes, the *demons* compute the current cost. Otherwise, the previously computed value is reported to the user without repeating the calculations.

9 Exergy Analysis

Exergy is defined as the work that is available in a gas, fluid or solid as a result of its nonequilibrium condition relative to some reference condition. Exergy losses are generated throughout the system by the irreversible change in entropy caused by the non-ideal performance that is inherent in all real systems. The purpose of performing exergy analysis is to identify the location, type and magnitude of the exergy losses in a system so that the system efficiency can be increased by minimizing these

[12]S. Potnis, Problem report for M.S in Computer Science, West Virginia University, 1988.

losses. In addition to estimating these losses, the factors that contribute to the losses must be considered. In some cases, complex and expensive design or modifications to the existing design may be necessary to minimize these losses; in other cases, the loss may be reduced by simply adding thermal insulation or sealing the leakages.

The basic equation for exergy calculation is the following:

$$\mathrm{Ex} = (h - h_0) - T_0(S - S_o)$$

The subscript "o" stands for a reference state, h is the enthalpy, T is the absolute temperature and S is the entropy. The above equation is used for each of the species involved in the combustion or chemical process. The total exergy is obtained by adding up the contributions from all the components.

Entropy and enthalpy of all the streams (input as well as output) at process conditions are taken from the report file. To obtain entropy and enthalpy of all these streams at reference conditions, each stream is considered as an input to a heater; the output from this heater is obtained at reference conditions. The values from the report file are used in the exergy calculations It should be noted that the exergy calculations are not very difficult but the analysis and interpretation of the results is often neither trivial nor straightforward and needs expertise in thermodynamics.

The exergy losses [13] for the Acrylic Acid flowsheet (Figure 15) is shown in Table 1.

TABLE 1

Exergy Losses in Acrylic Acid Plant

Submodel	Losses (Kcal/hr)	Kcal/Kmol of Output
Oxidation reactor	3.9E06	1650
Incinerator	4.4E06	2716
Extractor	3.9E05	227
Splitter	3.1E03	0.05
Mixer	3.1E04	21.78
Absorber1	2.6E06	39.95
Absorber2	7.99E03	5.65

From the tabulated values, it is apparent that the losses in the incinerator and the oxygen reactor are much higher compared to other submodels suggesting that these may be the areas needing improvement. In both cases, these losses can be classified under exergy loss resulting from transfer

[13] S. Shrotriya, Problem Report for M.S. in Computer Science, West Virginia University, 1991.

of chemical energy into heat energy. After studying the oxidation process, it becomes evident that the steam is primarily being used as a heat sink to keep the temperature low. Therefore, it may be suggested that steam be replaced by feed gases (labeled as *stack* in the flowsheet) to incinerator. These gases are burnt irreversibly in the incinerator.

After grouping the equipment under specific processes or types of loss (four such groups are identified [14]) the following table is constructed. These are the types of losses commonly encountered in a chemical process plant.

TABLE 2

Exergy Losses Commonly Encountered in a Chemical Process Plant

TYPE OF LOSS	CAUSE OF LOSS	SUGGESTED METHODS FOR REDUCING LOSS
1. Flue gas to atmosphere	Rejection of heat to atmosphere	Reduce rejection temperature Or Recover heat for industrial process
2. Combustion	Irreversible transformation of of chemical energy to heat energy	Inherent in the system, may be reduced by studying the process in detail.
3. Heat Transfer, Combustion gas	Heat transfer through temperature gradient	Raise steam temperature, improve heat transfer
4. Pump Efficiency	Friction, turbulence in fluid, bearing and seal friction	Improve component design

Exergy calculations were also performed for the Carbon Regeneration flowsheet. This is presented in Appendix D.

In summary, the exergy analysis module of IPSE performs the task of a specialist upto the extent of calculating the values using information in the ASPEN PCD and PDF, and providing general guidelines for interpreting the results.

10 THERMOPTER

ASPEN requires the user to specify the thermodynamic models (SYSOP0 - Ideal gas to SYSOP14 - Modified Redlich Kwong Soave) to be used to calculate various physical properties. Often the user does not have the knowledge of the key features of all the models provided in ASPEN to make a rational choice. The THERMOPTER (Thermodynamic Option Recommender) module[15]

[14] Ahern, J.E.,"The Exergy Method of Energy System Analysis", John Wiley & Sons.

[15] I.N. Balachandran - M.S. thesis, Chemical Engineering, West Virginia University, 1990. Also winner of the second place award at the AIChE student paper contest in Chicago, 1990.

was developed to assist the user in selecting the appropriate thermodynamic options based on the operating conditions in the ASPEN units, type of compounds present and other factors, such as numerical convergence properties, accuracy, etc. A literature survey on the previous work in this field indicated that the systems which were developed earlier to perform similar tasks are stand-alone systems and do not interface with any process simulator. The unique feature of THERMOPTER module is that it interfaces with ASPEN. Consequently, it accesses information from ASPEN's large knowledge base and the user is not required to provide every piece of information necessary to make a decision.

In IPSE, the information about the operating conditions and the types of compounds present in an ASPEN block are available in a structured form. Pure component critical properties and other relevant data are available in the ASPEN pure component data bank (PCD). Other necessary information is extracted from the user interactively. However, the user is always given the option of overriding the values provided by THERMOPTER.

LASER/NIE, the network inference engine subsystem of LASER was used to implement THERMOPTER. The inference networks in LASER/NIE consist of a combination of one or more relation links in IF-THEN form. Several pieces of evidence (the IF part) can lead to one hypothesis (the THEN part) . Also, one single piece of evidence can bear on several different hypotheses. Arbitrary logical combinations may be formed using the primitive operators such as AND, OR and NOT. Figure 19 shows a small hypothetical inference network of two hypotheses H1 and H2 influenced by two evidences E1 and E2. While E2 influences only H1, E1 influences both H1 and H2.

The strength of the influence of an evidence on a particular hypothesis is determined by two parameters LS and LN (Least Sufficiency and Least Necessity measures of the rule respectively), and is considered only when the evidence is present. These numbers are supplied by the domain expert. Each evidence has a certainty value ranging from -100 to 100. A certainty value of -100 indicates that the user is absolutely sure that the evidence is ABSENT, whereas a value of 100 implies that the user is absolutely sure that the evidence is PRESENT. A certainty value of 0 implies that there is no information available about the evidence.

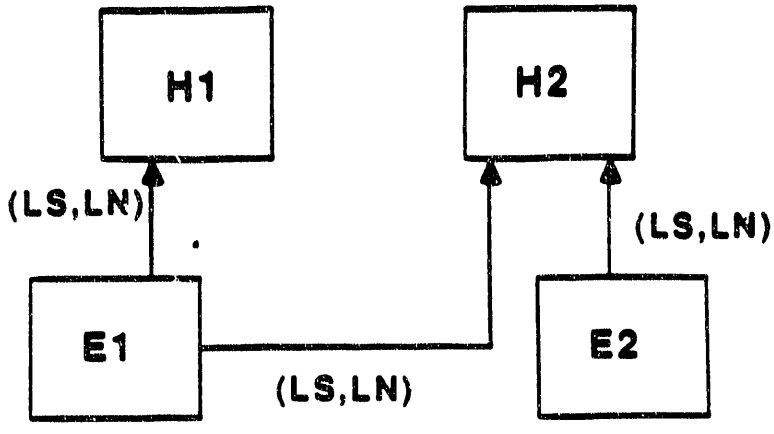


Figure 19 A Hypothetical Inference Network

10.1 The Inference Network

The inference network of THERMOPTER consisting of evidences and hypotheses is presented in Figure 20. All the *leaf* nodes were made "askable". This implies that the user can provide input when prompted by the system. The strength of the relationship between an evidence and the corresponding hypothesis node is determined by LS and LN parameters mentioned earlier which range from 0 to 10. LS is considered only when the evidence is present. An LS between 0 and 1 indicates that the presence of the evidence "discourages" the hypothesis. An LS between 1 and 10 indicates that the presence of the evidence "encourages" the hypothesis. Similarly, an LN between 0 and 1 indicates that absence of the evidence discourages the hypothesis, and an LN between 1 and 10 indicates that the absence of the evidence encourages the hypothesis.
There is no formal procedure to set these numbers. The system can be fine tuned only by testing with a variety of example problems. The probability values are propagated upwards, and the system will recommend the option with the highest probability.

Plausible relations (positively reinforcing and negatively reinforcing) constitute the majority of the relations. There are a few OR and AND relations in the network. Since a node is not allowed to be connected using more than one type of relation, "dummy" nodes were created to get around this limitation (Figure 20).

The prompts of each node in the inference network have a question attached to it. During an interactive session, these questions are asked to the user. Some of these questions are simple, such as *"Enter certainty for the presence of hydrogen"*. Others are more complex requiring deeper knowledge of thermodynamics, such as *"Enter the certainty for the presence of a wide boiling mixture"*.

10.2 Linking of THERMOPTER with ASPEN

Some of the nodes of the inference network were linked with the IPSE knowledge base that contains information about the ASPEN input file for which a thermodynamic package is to be recommended by THERMOPTER. For example, there is a rule in THERMOPTER that says *"If a liquid-liquid extractor is present, then do not use SYSOP8 and SYSOP8H"*. THERMOPTER searches through the IPSE knowledge base for an instance of the object EXTRC (corresponding to the same keyword in the ASPEN input language for a liquid-liquid extractor). If the search succeeds, the certainty of the node *"Extractor"* in the inference network is automatically set to 100 by THERMOPTER without user's intervention. Since the liquid-liquid extraction implies at least two liquid phases, the certainty of the node *"Phase splitting"* is also set to 100 by THERMOPTER.

In order to determine the certainty of the nodes *"Super-critical components"* and *"Critical-region"* in the inference network, the critical temperatures and pressures of various components are required. These have been retrieved from the ASPEN data banks and stored in a file for further processing.

THERMOPTER INFERENCE NETWORK

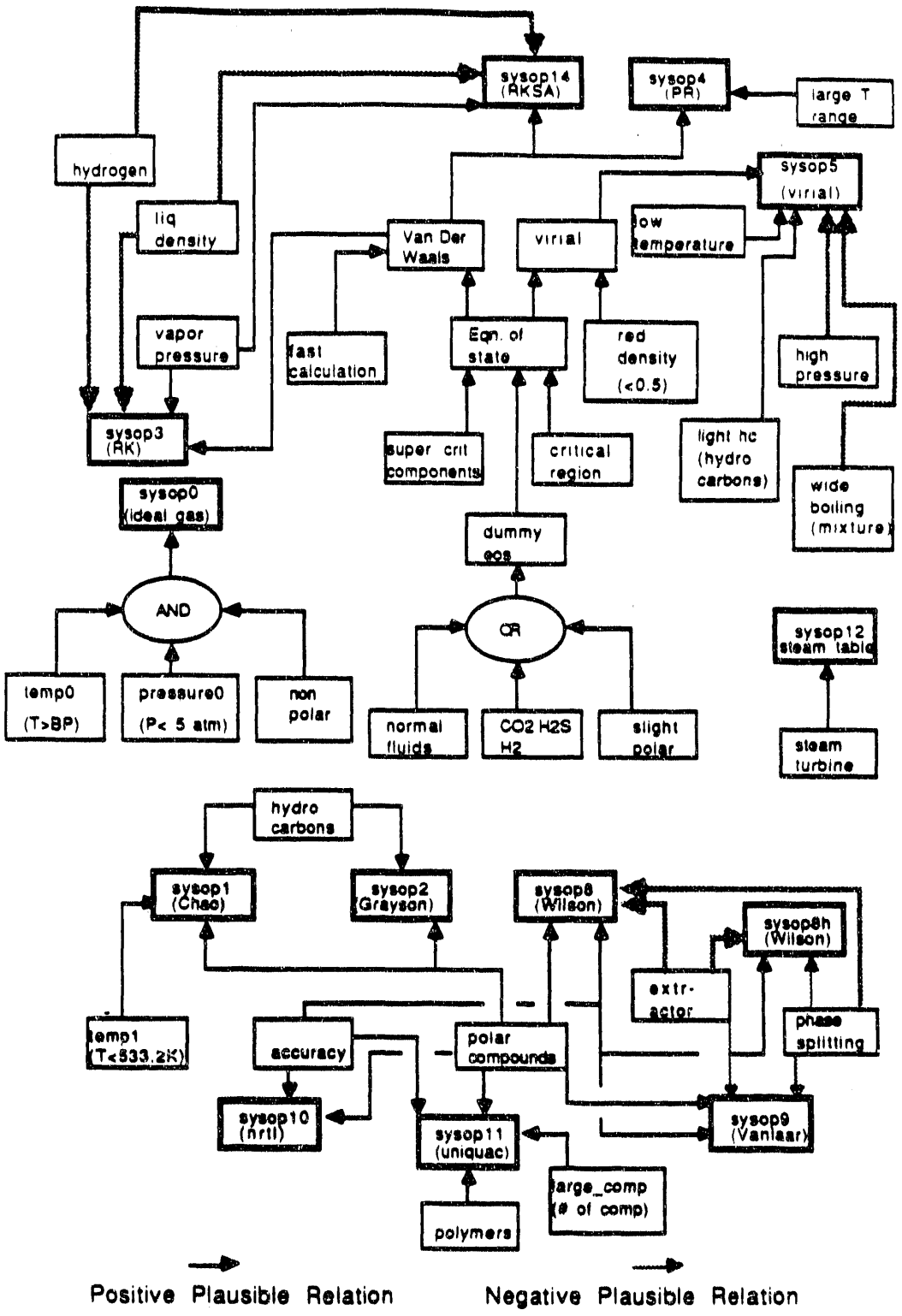


FIGURE 20

Since THERMOPTER interfaces with ASPEN, it performs the following tasks when needed:

1. Calculate the average reduced temperature
2. Predict the measure of polarity from the dipole moments
3. Recognize the type of ASPEN block
4. Recognize the presence of a certain compound
5. Retrieve the temperature of an ASPEN block.

10.3 Validation of THERMOPTER

THERMOPTER was validated[16] using fifteen test cases of wide ranges published in the Journal of Chemical Engineering Data. The results are shown in Table 3. In thirteen out of fifteen cases, THERMOPTER's recommendations matched identically with those of the authors. In the Test case 1, THERMOPTER's first choice turned out to be the author's second choice. In the test case 14, THERMOPTER recommended SYSOP8 whereas the authors recommended SYSOP3. The system for this test case consisted of Vapor-Liquid Equilibrium (VLE) of methane-methylal in the critical region. Methane is a nonpolar compound. However, methylal is a polar compound, and the mixture is therefore quite polar. Standard activity coefficient methods cannot model the system accurately because it is in the critical region. Standard equation of states cannot model the system accurately as the compound are polar. Since there are no rules or heuristics to help select the right model to estimate the VLE of polar mixtures in the critical region, THERMOPTER's recommendation for these systems may or may not be the best. Often the only recourse in this type of situation is to perform experiments and modify either a standard acitivity coefficient or a standard equation of state, such that it represents the data fairly accurately. However, THERMOPTER warns the user in such cases to pay special attention.

10.4 A Session with THERMOPTER

A step by step description of a typical vsession with THERMOPTER and the actions taken by it is included in Appendix C of this report. A list of the option sets provided by ASPEN is also included in Appendix C. THERMOPTER assumes that an ASPEN input file and the corresponding ASPEN PDF exist. The system is also capable of giving explanation of the recommendations.

[16] I.N. Balachandran - M.S. thesis , Chemical Enginnering, West Virginia University, 1990.

TABLE 3

THERMOPTER TEST CASES

Test case	System	Recommendations	
		Journal	THERMOPTER (belief)
1 *	VLE of ethane - propane at 255.4 K	SRK PR	PR SYSOP4 (20.0) BWR SYSOP5 (15.6) SRK SYSOP3 (15.1)
2	VLE of N2 - propane at 15.0 Mpa	PR	PR SYSOP4 (37.1)
3	VLE of ethyl acetate - cyclohexane - tetrahydrofuran at 313.15 K	Wilson NRTL	Wilson SYSOP8 (69.7) NRTL SYSOP10 (60.0)
4	LLE of water - tert-butyl alcohol - iso-butylene at 313.15 K	NRTL UNIQUAC	NRTL SYSOP10 (60.0) UNIQUAC SYSOP11 (58.9)
5	VLE of CO2 - cyclopentane at 318.15 K	PR	PR SYSOP4 (25.6)
6	LLE of acetonitrile - cyclohexane - benzene - acetone at 298.15 K	UNIQUAC	UNIQUAC SYSOP11 (62.3)
7	VLE of ethane - iso-butane at 356.75 K	PR	PR SYSOP4 (37.1)
8	VLE of methane - carbon-disulfide at 303.15 K	Wilson	Wilson SYSOP8 (62.6)
9	VLE of n-pentane - dichloro methane at 298.15 K	Wilson	Wilson SYSOP8 (62.6)
10	VLE of cyclohexane - tert-butyl alcohol at 328.2 K	Wilson	Wilson SYSOP8 (62.6)
11	VLE of n-pentane - ethanol at 372.7 K	Wilson	Wilson SYSOP8 (69.7)
12	LLE of water - ethanol - benzene at 303.15 K	UNIQUAC	UNIQUAC SYSOP11 (60.0)
13	VLE of methanol - ethanol - iso-propyl alcohol at 348.15 K	Wilson	Wilson SYSOP8 (69.7)
14 *	VLE of methane - methylal at 352.5 K	SRK*	Wilson SYSOP8 (62.6) SRK SYSOP3 (20.4)
15	VLE of ethanol - benzene - cyclohexane at 343.15 K	Wilson	Wilson SYSOP8 (62.6)

* SRK with non-classical mixing rules

11 Thermodynamic Data Checker: THERMOCHECK

In ASPEN, the thermodynamic models are grouped into 13 options. Each option contains a set of different models needed for vapor-liquid equilibrium data. Each of these options can be used to calculate basic properties, such as fugacity coefficient, enthalpy, entropy, molar volume etc. Each model requires specific parameters to be evaluated. As an example, the option set SYSOP10 (Renon) is shown in Table 4.

TABLE 4
OPTION SET SYSOP10 OF ASPEN

UNIVERSAL CONSTANT	PARAMETERS	
	UNARY	BINARY
TC PC VC ZC	PLXANT CPIG	RKTIG GMRENA
DHFORM DGFORM	RKTZRA VCRKT	GMRENB GMRENC
	DHLCVT	

Once an option set is selected by the user (either directly or from the recommendation by THERMOPTER), the user has to identify the models within the option needed to be evaluated. The next step is to identify the parameters required to solve the model by searching ASPEN PCD to locate the missing parameters. Often this requires a substantial amount of work. If the parameters are missing, ASPEN uses default values provided in the system during simulation. Frequently, this causes early failure when there are large number of missing parameters.

The purpose of THERMOCHECK[17] is to go through the ASPEN PCD to search for the existence (not the actual value) of relevant parameters for a model. Three different types of parameters are needed for thermodynamic equilibrium calculations:
(1) universal constants such as critical pressure, critical temperature,
(2) unary parameters such as specific heat, and
(3) model-specific binary parameters such as GMRENA in case of RENON (NRTL) model.

ASPEN PCD contains 23 different parameters for 300 compounds, but no binary parameters are available. Since THERMOCHECK needs to check the existence of only a few parameters, a shorter version of ASPEN PCD was created for this purpose. The creation of this version is justified because in a typical flowsheet, there are several streams containing various components, each requiring several properties to be evaluated. This makes the manual search through the ASPEN PCD very tedious. A C program (*stripper.c*) was written to create the shorter version from the original huge data bank in a "bit vector representation". This facilitates an efficient search for the existence of the parameters for each compound. An "on" bit in a certain position for a compound indicates the presence of the parameter.

[17] S. Shrotriya, Problem Report for M.S. in Computer Science, West Virginia University, 1991.

11.1 Binary Parameters

Since ASPEN does not contain these parameters, these are to be provided by the user. Following the ASPEN technical reference manual and other technical sources, THERMOCHECK provides on-line help on the required binary parameters and their default values. Help is also provided on the use of ASPEN's regression method for evaluating the parameters from experimental data. As an example, the evaluation of Wilson binary parameters is provided in the source directory of THERMOCHECK. Similar regression techniques can be used to establish other model parameters that are missing.

12 IKAM: IPSE Knowledge Acquisition Module

This module was initially designed (and referred to as the *rule editor* in the earlier documents) to capture only the problem solving knowledge in the form of rules that are developed at the equipment block level. However, the scope of the module was expanded to capture other forms of knowledge, such as knowledge about various equipment blocks (submodels) that were not included in the existing IPSE knowledge base. In either case the relevant information is captured from the user in an interactive session. The information is subsequently transformed by IKAM into *objects* as required by the system.

IKAM is also capable of capturing information about rules regarding compositional changes. It also enables the users to formulate rules in the *grouped* mode as well as for specific key components.

To further the capabilities of IKAM, it was decided to incorporate the knowledge about equilibrium reactions (since gasification problems often contain equilibrium reactions) using Le Chatelier's principle. As a result, IKAM will prompt for relevant/required information from the user to develop rules.

The current IKAM facilitates rule creation for the IPSE analyzer that uses LASER/RPS. It can be extended to perform knowledge acquisition for new rules for THERMOPTER to build an inference network by LASER/NIE.

13 IPSE Systems Manual

The manual describes the directory structure, contents of all the files, and purpose of the routines. The intended users of this document are the programmers who want to extend the system, or re-use the existing modules in the system to perform similar tasks. The directory structure of IPSE is shown in Figure 21.

14 IPSE Users Manual

The IPSE users manual contains necessary information to use the IPSE system. It is assumed that the users have familiarity with the ASPEN simulator and various chemical engineering processes.

Knowledge available in the IPSE knowledge base to use various modules of IPSE is presented in Figure 22.

The device DISK$IPSE contains directory [ipse]. The directory structure is as shown below.

The Directory Structure of IPSE

```
                              DISK$IPSE:[IPSE]
                                     |
_____________________________________________________________________
   |          |          |          |          |                  |
[ASPEN]    [DOC]      [KB]      [LIB]      [REF]              [SOR]
                                             |                  |
_____________________________________________|                  |
|              |              |              |                  |
|-[COST]       |-[GRAPHIC]    |-[SCANNER]    |-[COMPOSITION_RULE] |
|-[FLEXI]      |-[METHODS]    |-[SDFSOR]     |-[DOMAIN]           |
|-[FRONT]      |-[RGROUP]     |-[SOR]        |-[WRITE_RULE]       |
|-[GENERIC]    |-[RULEDT]     |-[SUBMODEL]                       |
|-[GOALEDT]    |-[RULES]      |-[SUPPORT]                        |
                                                                 |
_____________________________________________________________________|
|              |              |                        |
|-[COST]       |-[GOALEDT]    |-[RULEDT]-------|       |-[SUBMODEL]
|-[FLEXI]      |-[GRAPHIC]    |-[RULES]        |       |-[SUPPORT]
|-[FRONT]      |-[METHODS]    |-[SCANNER]      |
|-[GENERIC]    |-[RGROUP]     |-[SDFSOR]       |
                                              |
                                              |
_____________________________________________|
|                             |              |
|-[COMPOSITION_RULE]          |-[DOMAIN]     |-[WRITE_RULE]
```

Each of the sub directory of sor and ruledt has a sub directory (not shown) called as [CMSLIB]. This sub directory contains the Vax Code Management System library files for its directory. The sub directories of ref are the reference copy directories of all CMS sub directories of the ipse project. This means that they contain the files of the last generation replaced or reserved.

Figure 21

IPSE Module \ Available Knowledge	1 Generic Submodels	2 ASPEN SDF	3 ASPEN PDF	4 1 & 3	5 Submodel Instances	6 Model Goals	7 RPS Rules
Submodel Editor	▼	△	△	●	●	●	●
Rule Editor	▼	△	△	●	●	●	●
PRE Analyzer	○	○	○	○	○	▼	●
Thermopter	△	○	▼	●	●	●	●
Thermocheck	△	○	▼	●	●	●	●
Submodel Instantiator	○	○	○	▼	⊗	⊗	⊗
Economic Analyzer	○	○	○	○	▼	●	●
Goal Editor	○	○	○	○	▼	●	●
RPS Analyzer	○	○	○	○	○	○	▼
Exergy Analyzer	○	○	○	○	▼	●	●

KEY:

▼ **Necessary & Sufficient** △ **Not Necessary** ○ **Necessary But Not Sufficient** ● **More Than Sufficient** ⊗ **Not Appropriate**

Figure 22 Available Knowledge for Using IPSE modules

15 PROJECT PERSONNEL

1. **Dr. Richard Turton** - Associate Professor, Chemical Engineering, West Virginia University.
2. **Shailesh Potnis** - M.S. (Chemistry), M.S. (Computer Science) - Research Associate, Dept. of Computer Science, West Virginia University.
3. **I.N. Balachandran** - M.S.ChE. West Virginia University.
4. **Sachin Shrotriya** - M.S.ChE, MS (Computer Science), West Virginia University.
5. **Devendra Godbole** - MS (Mechanical Engg), M.S. (Computer Science), West Virginia University.
6. **Dr. Sumitra Reddy - Principal Investigator** - Assistant Professor, Computer Science, West Virginia University.

16 APPENDIX A: IKAM for Rules

The following script was generated using the new rule creation facilities of IKAM. The rules pertain to the submodel *Char-gasifier*. From the user's input, IPSE creates the rule object which is also included at the end of the script. To demonstrate how the newly created rule was used by the analyzer to meet a specified goal, a script following the rule creation is also included.

```
    Would you like the graphics mode ? yes no
    Make an appropriate choice <n>

  Enter rule file or quit for quit  [newrules.c]  <CR>

    create a new rule ? yes no
    make an appropriate choice <y>

    available submodels for making rules...

    absorber
    scrubber
    dryer
    activator
    after_burner
    gas_generator
    char_gasifier
    distillation_column
    extractor

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

    stream_heater
    flow_mixer
    oxidation_reactor
    separator
    flow_splitter
    previous menu
    quit

    Select the submodel for making the rule  <char>

    The possible goal streams of char_gasifier are...

    outlet
    previous menu
    quit

    select the goal stream for this rule <out>

    The attributes for char_gasifier's outlet
```

```
   temperature
   pressure
   flow
   key_component
   composition

   select the goal attribute <compos>

please enter the goal component  []  <carbon_monoxide>

composition_direction of the char_gasifier's outlet is to be

   increased
   decreased

   select an appropriate goal operator  <inc>

   modify char_gasifier's input streams to meet the goal ? yes no

   make an appropriate choice  <y>

   The possible streams of char_gasifier are

   feed_stream
   inlet
   previous menu
   quit

   select an appropriate stream to meet the goal  <feed>

   The attributes for char_gasifier's feed_stream

   temperature
   pressure
   flow
   key_component
   composition

   select the action_stream attribute  <compos>

   please enter the goal component  []   <water>

   direction of composition_direction of feed_stream

   increase
   decrease

   Select the direction to meet the goal   <dec>
```

```
The possible streams of char_gasifier are

feed_stream
inlet
previous menu
quit

select an appropriate stream to meet the goal  <prev>

modify char_gasifier's attributes to meet the goal ? yes no

make an appropriate choice  <y>

The available properties of char_gasifier are

pressure_direction
temperature_direction
previous menu
quit

Make an appropriate choice   <tempe>

direction of temperature_direction of char_gasifier

increase
decrease

Select the direction to meet the goal   <inc>

The available properties of char_gasifier are

pressure_direction
temperature_direction
previous menu
quit

Make an appropriate choice    <prev>

create more rules ?  yes  no
make an appropriate choice   <n>
```

The following script was generated to demonstrate how the newly created rule was successfully used by the IPSE analyzer to meet a goal.

```
Would you like the graphics mode ?  yes   no
Make an appropriate choice   <n>
```

```
 enter the pdf file name  []   <csto>

Enter the type of goal

 add goals
 delete goals
 previous menu
 quit

 make an appropriate choice   <add>

Goals can be set on the following streams

 s16
 s6
 s7
 s4
 s2
 s5

Select an appropriate stream as a goal stream <16>
 The attributes for s16 are

 temperature
 pressure
 total_flow
 composition

 Select an appropriate attribute   <compos>

The components and their mole fractions of s16 are...

 c (carbon) 0
 ch4 (methane) 1.131220e-03
 co (carbon_monoxide) 0.262569
 co2 (carbon_dioxide) 6.322616e-02
 h2 (hydrogen) 5.983250e-02
 h2o (water) 9.829850e-02
 h2s (hydrogen_sulfide) 1.131220e-03
 n2 (nitrogen) 0.4714
 o2 (oxygen) 4.241777e-02

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

 s (sulfur) 0
 select an appropriate key component as goal  <mono>

possible operators for the mole fraction of s16's co are
```

```
 equal to
 greater tha
 less than

 The current value is 0.262569 ...

 <gre>

the current mole fraction of s16's co is 0.262569
the mole fraction of s16's co should be greater than....
enter the goal value   []     <0.3>

 Enter the type of goal

 add goals
 delete goals
 previous menu
 quit

 make an appropriate choice    <q>

 s16's co (carbon_monoxide) mole fraction should be greater than 0.3
The actual value is  0.262569
Hence this goal is not satisfied

IPSE analysis begins
 Do you want the IPSE explanation facility during the IPSE analysis?

   yes
   no
   help

Make an appropriate choice   <n>

To increase the co (carbon_monoxide) content of
  outlet s16 of char_gasifier char_gasifier_1
  follow the following recommendations

  1) decrease the water, h2o content of the feed_stream s7
  2) increase the temperature of the char_gasifier char_gasifier_1

  To decrease the h2o of the outlet stream s7,
  of flow_mixer_2 follow one or more of the following recommendations

  1) decrease the h2o content of the inlet stream s6
  2) decrease the h2o content of the inlet stream s4
```

To decrease the h2o of the outlet stream s6,
of flow_mixer_1 follow one or more of the following recommendations

1) decrease the h2o content of the inlet stream s5

To decrease the h2o of the outlet stream s4,
of flow_mixer_3 follow one or more of the following recommendations

1) decrease the h2o content of the inlet stream s2

To decrease the h2o of the outlet stream s5,
of flow_splitter_1 follow the following recommendation

1) decrease the flow rate of the inlet stream s1

To decrease the h2o of the outlet stream s2,
of flow_splitter_1 follow the following recommendation

1) decrease the flow rate of the inlet stream s1

The rule objects created by IPSE from the user-supplied information are given in Figure A1 and Figure A2.

```
{char_gasifier_rule_group
   instanceof # :

 LSR_GLOBAL_RULE_GROUP
  SUB_RULES * (SUB_RULE_OF) :
   char_gasifier_rule_2
   char_gasifier_rule_3
   char_gasifier_rule_4
   char_gasifier_rule_5
   char_gasifier_rule_6
   char_gasifier_rule_7
   char_gasifier_rule_8
   control_rule_2
   control_rule_3
   char_gasifier_rule_665747888
  rule_group_of * (rule_group) : char_gasifier
  COMMENT : "This rule group contains the rules and/or rule groups for\n
 the submodel char_gasifier.\n"
         "This rule group is selected when the goal_stream ?goal_stream\n"
 "is an outlet of the submodel ?cur_sub which is an instance"
         "of the generic submodel char_gasifier"
 CONTROL : "goal goal_stream = ?goal_stream"
 ENTRY_ACTION : "lsr_setval goal analysis_mode 'FOCUSSED'"
                "call deactivate_rule_groups"
 EXIT_ACTION : "lsr_setval goal analysis_mode 'GLOBAL'"
               "call activate_rule_groups"
               "call explain_ipse_analysis"
 LHS : "goal analysis_mode = 'GLOBAL'"
       "goal goal_stream = ?goal_stream"
       "?goal_stream inlet = ?cur_sub"
       "?cur_sub instanceof = 'char_gasifier'"
 PRIORITY : 20
}
```

Figure A1
A Rule Group Object cCeated by IPSE/IKAM from User's Input

```
{ char_gasifier_rule_665747888
   instanceof # : LSR_LOCAL_RULE
   SUB_RULE_OF * (SUB_RULES) : char_gasifier_rule_group
    LHS :  "goal goal_stream = ?outlet"
           "?outlet key_component = ?key_component"
           "?outlet tested_key_component ][ ?key_component"
           "?outlet composition_direction = 'increase'"
           "?outlet generic_key_component = 'carbon_monoxide'"
    "?outlet inlet = ?char_gasifier"
    "?char_gasifier feed_stream = ?feed_stream"
    RHS : "lsr_addval ?outlet tested_key_component ?key_component"
   "write 'To increase the ?key_component
              (carbon_monoxide) content of\n'"
    "write 'outlet ?outlet of char_gasifier ?char_gasifier\n'"
    "write 'follow the following recommendations\n\n'"
          "lsr_addval ?feed_stream composition_direction 'decrease'"
          "lsr_addval goal goal_stream ?feed_stream"
          "lsr_addval ?feed_stream generic_key_component 'water'"
          "call set_component_id {?retval ?feed_stream}"
          "write ' 1) decrease the water, ?retval content of the
           feed_stream ?feed_stream\n\n'"
            "lsr_addval stream_table stream_name ?feed_stream"
            "lsr_addval stream_table direction 'decrease'"
           "lsr_addval ?char_gasifier temperature_direction 'increase'"
           "lsr_addval submodel_table submodel_name ?char_gasifier"
           "lsr_addval submodel_table attribute 'temperature_direction'"
           "lsr_addval submodel_table direction 'increase'"
           "write '   2) increase the temperature of the char_gasifier
           ?char_gasifier\n'"
           "lsr_popval ?outlet key_component"
           "lsr_popval ?outlet composition_direction"
           "lsr_popval ?outlet generic_key_component"
           "lsr_popval goal goal_stream"
    }
```

Figure A2
A rule object created by IPSE/IKAM from user's input

The C program generated by IKAM to create the rule object is given below.

```
      #include <laserkr.h>
      #include <laserrps.h>
char *rule, *l = "LHS", *r = "RHS", *so = "SUB_RULE_OF", *c = "COMMENT";

       char_gasifier_rule_665747888 () {
        lsr_makeobj (rule = "char_gasifier_rule_665747888");
        lsr_relate (rule, "instanceof", "LSR_LOCAL_RULE");
        lsr_relate (rule, so, "char_gasifier_rule_group");
        lsr_addval (rule, l,
        LSR_STR ("goal goal_stream = ?outlet"));
        lsr_addval (rule, l,
        LSR_STR ("?outlet key_component = ?key_component"));
        lsr_addval (rule, l,
        LSR_STR ("?outlet tested_key_component ][ ?key_component"));
        lsr_addval (rule, l,
        LSR_STR ("?outlet composition_direction = 'increase'"));
        lsr_addval (rule, l,
        LSR_STR ("?outlet generic_key_component = 'carbon_monoxide'"));
        lsr_addval (rule, l,
        LSR_STR ("?outlet inlet = ?char_gasifier"));
        lsr_addval (rule, l,
        LSR_STR ("?char_gasifier feed_stream = ?feed_stream"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval ?outlet tested_key_component ?key_component"));
        lsr_addval (rule, r,
        LSR_STR ("write '\tTo increase the ?key_component
                 (carbon_monoxide) \content of\n'"));
        lsr_addval (rule, r,
        LSR_STR ("write '\toutlet ?outlet of char_gasifier ?char_gasifier\n'"));
        lsr_addval (rule, r,
        LSR_STR ("write '\tfollow the following recommendations\n\n'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval ?feed_stream composition_direction 'decrease'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval goal goal_stream ?feed_stream"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval ?feed_stream generic_key_component 'water'"));
        lsr_addval (rule, r,
        LSR_STR ("call set_component_id {?retval ?feed_stream}"));
        lsr_addval (rule, r,
        LSR_STR ("write '\t1) decrease the water, ?retval content of \
                the feed_stream ?feed_stream\n\n'"));
        lsr_addval (rule, r,
LSR_STR ("lsr_addval stream_table stream_name ?feed_stream"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval stream_table direction 'decrease'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval ?char_gasifier temperature_direction 'increase'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval submodel_table submodel_name ?char_gasifier"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval submodel_table attribute \
```

```
                'temperature_direction'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_addval submodel_table direction 'increase'"));
        lsr_addval (rule, r,
        LSR_STR ("write '\t2) increase the temperature of \
                the char_gasifier ?char_gasifier\n'"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_popval ?outlet key_component"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_popval ?outlet composition_direction"));
        lsr_addval (rule, r,
        LSR_STR ("lsr_popval ?outlet generic_key_component"));
lsr_addval (rule, r,
        LSR_STR ("lsr_popval goal goal_stream"));
     }
```

The following C program (user_created_rules.c) calls the above C program to generate the rule objects.

```
     user_created_rules (subname)
char *subname; {
if ((!strcmp (subname, "char_gasifier")) &&
(!lsr_isobj ("char_gasifier_rule_665747888")))
  char_gasifier_rule_665747888 ();
     }
```

17 APPENDIX B: IKAM for Submodels

The following script is generated by the IKAM module of IPSE for submodel creation for a new equipment block.

```
WELCOME  TO   IPSE
INTELLIGENT PROCESS SIMULATION ENVIRONMENT
PROGRAMMING ENVIRONMENT : LASER
SIMULATION ENGINE : ASPEN

Would you like the graphics mode ?   yes  no
Make an appropriate choice    n

Do you wish to create additional submodels ?

yes
no
help
answer yes or no or help for help   y

enter a unique name for the submodel to be created or enter help
  to get the listing of the existing submodels  [help]  <CR>

  IPSE Knowledge Acquisition Module (IKAM) provides to the users the
  facility called as Submodel Editor.  This facility enables the users
 to interactively add additional submodels or the physical equipment
  blocks to the IPSE knowledge base.  If the  equipment blocks
 such as char_gasifier or absorber etc which are already present in
 the IPSE knowledge base are not sufficient or appropriate for the
 types of the simulation models which are normally delt with at your
 site, then you may add the descriptive knowledge of the additional
 equipments using this submodel editor facility and then add the
 behavioral knowledge of these additional submodels through the rule
 editor facility of IKAM.
 The following submodels are already in the knowledge base

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

 The following submodels exist in the knowledge base

  absorber
  scrubber
  dryer
  activator
  after_burner
  gas_generator
  char_gasifier
  distillation_column
  stream_heater

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

  flow_mixer
  oxidation_reactor
  separator
  flow_splitter
```

```
** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

 enter a unique name for the submodel to be created or enter help
  to get the listing of the existing submodels  [help]  extractor

  Are there any output streams of extractor
  such that goals can be set on their properties?

  yes
  no
  help help

  You may identify the characteristic output streams of the submodel
  which is being developed, such that the goals can be set on their
  attributes.

 For example

  Consider the submodel "char_gasifier" which has"gas_output_stream"
  as one of the characteristic output streams such that it is possible
  to set goals on its temperature, composition or flow rate.

  ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

 I can suggest some of the possible output streams of extractor
  on whose properties goal may be set:

 outlet
 composition_outlet
 gas_output_stream
 set_flow_stream
 variable_flow_stream
 extract_stream
 raffinate_stream
 scrubbed_stream
 solution_stream

 ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

  top_stream
  bottom_stream
  none

 Select one of the attributes listed above or Select "none"
 to input some other attribute for "extractor".   extract_stream

 Are there any other output streams of extractor
 such that goals can be set on their properties?

 yes
 no
 help    y
```

```
 Enter the characteristic output stream
 for extractor -   []   raffinate_stream

Are there any other output streams of extractor
such that goals can be set on their properties?

 yes
 no
 help   n

Are there any input streams of extractor
Which may influence any of its goal stream(s)?

 yes
 no
 help    help

You may identify the characteristic input streams of the physical
equipment or the submodel, which you are in the process of creating,
and which can influence one or more attributes of the possible goal
streams.
For example:

Consider submod absorber" which has the "absorbent_stream" as
one of the characteristic input stream such that it is possible to
influence the "key component" composition of either the
 "scrubbed_stream" or the "solution_stream" streams which are the
  goal streams of the submodel "absorber".

 ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

I can suggest some of the possible input streams of extractor
whose properties may influence the output streams:

 inlet
 composition_inlet
 gas_input_stream
 feed_stream
 solvent_stream
 absorbent_stream
 air_stream
 steam_stream
 fuel_stream

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

 none
Select one of the attributes listed above or Select "none"
to input some other attribute for "extractor".   feed_stream

Are there any other input streams of extractor
Which may influence any of its goal stream(s)?

yes
```

```
 no
 help    y

Enter the characteristic input stream
 for extractor -   []  solvent_stream

 Are there any other input streams of extractor
  Which may influence any of its goal stream(s)?

  yes
  no
  help   n

Are there any attributes of extractor
 Which may influence any of its goal stream(s)?

 yes
 no
 help   help

You may identify the characteristic attributes of the physical
equipment or the submodel, which you are in the process of creating,
and which can influence one or more attributes of the possible goal
streams.

For example:

 Consider submodel distilation_column" which has "number of stages"
 as one of the characteristic attribute such that it is possible to
 influence the "key component" composition of either the "top"
 or the "bottom" streams which are the attributes of the goal
 streams of the  submodel "distillation column".

 ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

I can suggest some of the possible properties of extractor
   whose properties may influence the output streams:

   pressure_direction
   temperature_direction
   no_stages_direction
   none

  Select one of the attributes listed above or Select "none"
   to input some other attribute for "extractor". none

   Are there any other attributes of extractor
   Which may influence any of its goal stream(s)?

   yes
   no
   help   no

    Do you wish to create additional submodels ?
```

```
yes
no
help
answer yes or no or help for help   n

Should I write the knowledge base to a file ?  yes  no
Make an appropriate choice   y

Enter the knowledge base file [
  IPSEKB:SUBMODEL.KB]  subedt.kb

pr extractor


{ extractor
   isa # : sub_model
    extract_stream <meta_extract_stream> :
    feed_stream <meta_feed_stream> :
    raffinate_stream <meta_raffinate_stream> :
    solvent_stream <meta_solvent_stream> :
}


pr meta_extract_stream

{ meta_extract_stream
   isa # : meta_information
    contents : "The  extract stream is "
    goal_attribute : "standard"
    goal_stream :
    method : set_specific_outlet 2
 }

pr meta_feed_stream

{ meta_feed_stream
   isa # : meta_information
   action_attribute : "standard"
   action_stream :
   contents : "The feed stream is "
   method : set_specific_inlet 2
}

quit

    Should IPSE continue analysis?

    yes
    no
    help

    Make an appropriate choice    n
```

18 APPENDIX C: THERMOPTER Session

The option sets in ASPEN are given in Table C1.

A session with THERMOPTER module to get advice on the selection of a thermodynamic option consists of the following steps.

1. THERMOPTER asks for the name of the input file
2. At this point IPSE creates a knowledge base called *.kb where * stands for the first four letters of the input file. THERMOPTER sets the certainty values of the relevant nodes to appropriate numbers as default values.
3. THERMOPTER prompts the user for certainty values of the nodes. During this step, the user can override the certainty values set by THERMOPTER by entering values of his/her choice. If the user has no answer to a question, it can be skipped and the system uses 0 as the certainty value in this case. However, the greater the number of answers given, the more accurate will be the recommendation.
4. THERMOPTER evaluates the "belief" of all the hypotheses (ASPEN has 13 of these), using the certainty values, and recommends the hypothesis (a thermodynamic option) with the highest "belief". Internally, Bayesian statistics are used by LASER/NIE to determine the belief from the certainty values of the evidences. All the other hypotheses are presented in descending order of belief.

TABLE C1

THERMODYNAMIC OPTION SETS IN ASPEN

OPTION	FUGACITY COEFFICIENT	
	VAPOR PHASE	LIQUID PHASE
SYSOP0	Ideal Gas	Ideal Liquid
SYSOP1	Redlich -Kwong	Chao-Seader
SYSOP2	Redlich -Kwong	Grayson-Streed
SYSOP3	Redlich-Kwong-Soave	Redlich-Kwong-Soave
SYSOP4	Peng-Robinson	Peng-Robinson
SYSOP5	Benedict-Webb-Rubin	Benedict-Webb-Rubin
SYSOP8	Redlich -Kwong	Wilson
SYSOP8H	Redlich -Kwong	Wilson-Henry's law
SYSOP9	Redlich -Kwong	Van Laar
SYSOP10	Redlich -Kwong	NRTL
SYSOP11	Redlich -Kwong	UNIQUAC
SYSOP12	ASME Steam Tables	ASME Steam Tables
SYSOP14	Redlich-Kwong-Soave-ASPEN	Redlich-Kwong-Soave-ASPEN

19 APPENDIX D : Exergy Calculations

TABLE D1

Exergy values for carbon-regeneration plant.

UNIT NAME	EXERGY(Btu/hr)	EXERGY(Btu/lbmol)
Activator_4	164654.14	696.29
Flow_mixer_5	279010.33	312.97
Gas_generator_2	1675379.14	7132.24
Dryer_3	2592374.66	3804.99
Gas_generator_1	3009872.58	7132.24
After_burner_6	6760339.96	4480.77
Scrubber_7	9988128.90	3212.91

20 APPENDIX E: Script for Explanation

The following script was generated from invoking the *explanation* facilities of the IPSE analyzer.

```
        WELCOME  TO   IPSE
        INTELLIGENT PROCESS SIMULATION ENVIRONMENT
        PROGRAMMING ENVIRONMENT : LASER
        SIMULATION ENGINE : ASPEN

      Would you like the graphics mode ? yes  no
Make an appropriate choice   <n>

      have you already grouped the ASPEN blocks in to submodels
      and saved the knowledge base?
      yes
      no
      help

      Make an appropriate choice   <y>

    please enter the submodel knowledge base file  []   char_gasifier.kb

      Do you wish to use THERMOPTER

      yes
      no
      help

      <no>

       Do you wish to use THERMOCHECK
       yes
       no
      help

      <no>

      Do you wish to add/modify goals ?

      yes
      no

       Make an appropriate choice

       <y>

       Enter the type of goal
```

```
add goals
 delete goals
 previous menu
 quit

make an appropriate choice

 <add>

Goals can be set on the following streams

 s16
 s6
 s7
 s4
 s2
 s5

Select an appropriate stream as a goal stream

  <16>

 The attributes for s16 are

 temperature
  pressure
 total_flow
 composition

 Select an appropriate attribute

   <compos>

The components and their mole fractions of s16 are...

  c (carbon) 0
 ch4 (methane) 1.131220e-03
 co (carbon_monoxide) 0.262569
co2 (carbon_dioxide) 6.322616e-02
h2 (hydrogen) 5.983250e-02
h2o (water) 9.829850e-02
 h2s (hydrogen_sulfide) 1.131220e-03
  n2 (nitrogen) 0.4714
  o2 (oxygen) 4.241777e-02

  ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]
```

```
s (sulfur) 0
select an appropriate key component as goal

 <monox>

 possible operators for the mole fraction of s16's co are

 equal to
 greater than
 less than

 The current value is 0.262569 ...

   <less>

 the current mole fraction of s16's co is 0.262569
 the mole fraction of s16's co should be less than....
   enter the goal value    []    <0.2>

 Enter the type of goal

 add goals
 delete goals
 previous menu
 quit

make an appropriate choice

 <q>

Should I write the knowledge base to a file ?

 yes
 no

Make an appropriate choice <n>

Do you wish to proceed with the RPS analysis ?

  yes
  no
 Make an appropriate choice
  <y>

s16's co (carbon_monoxide) mole fraction should be less than 0.2
 The actual value is  0.262569
 Hence this goal is not satisfied
IPSE analysis begins
```

```
Do you want the IPSE explanation facility during the IPSE analysis?

 yes
 no
help

 Make an appropriate choice

  <y>

 To decrease the co of the outlet stream
s16 of the char_gasifier char_gasifier_1, follow one or more o
the following recommendations:

  1) increase the inlet stream s7 h2o content
  2) decrease the temperature of the char_gasifier_1.

  Should IPSE explain how the analysis was carried
  out this far?

   yes
   no
   help

   make an appropriate choice
   <y>

    Please enter the number of the last rules which were
    fired to be explained.  Enter "all" if you desire the explanation of
    all the rules which were fired.    [all]

   Explanation of the all previous rules follows

1. This rule group contains the rules and/or rule groups for
    the submodel char_gasifier.
    This rule group is selected when the goal_stream s16
    is an outlet of the submodel char_gasifier_1 which is an
    instance of the generic submodel char_gasifier

2. To decrease the carbon monoxide, co content  of the
    outlet stream, s16 of the char_gasifier char_gasifier_1,
    decrease the temperature of the char_gasifier_1 and
    increase the flow rate of steam in the inlet stream s7
The explanation of the all previous rules is complete

  ** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

    To increase the h2o of the outlet stream s7,
```

```
of flow_mixer_2 follow one or more of the following recommendations

1) increase the h2o content of the inlet stream s6
2) increase the h2o content of the inlet stream s4

Should IPSE explain how the analysis was carried
out this far?

yes
no
help

make an appropriate choice
<no>

To increase the h2o of the outlet stream s6,
of flow_mixer_1 follow one or more of the following recommendations

1) increase the h2o content of the inlet stream s5

Should IPSE explain how the analysis was carried
out this far?

yes
no
help

make an appropriate choice

<no>

To increase the h2o of the outlet stream s4,
of flow_mixer_3 follow one or more of the following recommendations

1) increase the h2o content of the inlet stream s2
Should IPSE explain how the analysis was carried
out this far?

yes
no
help

make an appropriate choice

<no>

To increase the h2o of the outlet stream s5,
of flow_splitter_1 follow the following recommendation
```

```
1) increase the flow rate of the inlet stream s1

Should IPSE explain how the analysis was carried
out this far?

 yes
 no
 help

make an appropriate choice

 <y>

 Please enter the number of the last rules which were
fired to be explained.  Enter "all" if you desire the explanation of
all the rules which were fired.    [all]  <2>

Explanation of the 2 previous rules follows

9. This rule group contains the rules and/or rule groups for
    the submodel flow_splitter

    This rule group is selected when the goal_stream s5
    is an outlet of the submodel flow_splitter_1 which is an
    instance
    of the generic submodel flow_splitter

10. To increase the key_component h2o content of the output
    stream s5 of the flow_splitter flow_splitter_1, call the
    appropriate routine attached to the key_component_increase
    property of the flow_splitter flow_splitter_1

The explanation of the 2 previous rules is complete

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

To increase the h2o of the outlet stream s2,
of flow_splitter_1 follow the following recommendation
 1) increase the flow rate of the inlet stream s1

Should IPSE explain how the analysis was carried
 out this far?

 yes
 no
 help
```

```
make an appropriate choice
 <y>

 Please enter the number of the last rules which were
 fired to be explained.  Enter "all" if you desire the explanation of
 all the rules which were fired.    [all]

 Explanation of the all previous rules follows

1. This rule group contains the rules and/or rule groups for
    the submodel char_gasifier.
    This rule group is selected when the goal_stream s16
    is an outlet of the submodel char_gasifier_1 which is an
    instance of the generic submodel char_gasifier

2. To decrease the carbon monoxide, co content of the
    outlet stream, s16 of the char_gasifier char_gasifier_1,
    decrease the temperature of the char_gasifier_1 and
    increase the flow rate of steam in the inlet stream s7

3. This rule group contains the rules and/or rule groups for
    the submodel flow_mixer

This rule group is selected when the goal_stream s7

 Continue?
  (Y or N or C)[Y]
 is an outlet of the submodel flow_mixer_2 which is an instance
    of the generic submodel flow_mixer

4. To increase the key_component content of the output stream s7
    of the flow_mixer flow_mixer_2, increase the inlet key_compo-
    nent contents by calling the appropriate method attached to the
    temp_increase property of the flow_mixer flow_mixer_2

5. This rule group contains the rules and/or rule groups for
            the submodel flow_mixer

    This rule group is selected when the goal_stream s6
    is an outlet of the submodel flow_mixer_1 which is an
    instance of the generic submodel flow_mixer

 Continue?
 (Y or N or C)[Y]

6. To increase the key_component content of the output stream s6
    of the flow_mixer flow_mixer_1, increase the inlet key_compo-
    nent contents by calling the appropriate method attached to the
```

temp_increase property of the flow_mixer flow_mixer_1

7. This rule group contains the rules and/or rule groups for the submodel flow_mixer

This rule group is selected when the goal_stream s4 is an outlet of the submodel flow_mixer_3 which is an instance of the generic submodel flow_mixer

8. To increase the key_component content of the output stream s4

Continue?
(Y or N or C)[Y]

of the flow_mixer flow_mixer_3, increase the inlet key_component contents by calling the appropriate method attached to the temp_increase property of the flow_mixer flow_mixer_3

9. This rule group contains the rules and/or rule groups for the submodel flow_splitter

This rule group is selected when the goal_stream s5 is an outlet of the submodel flow_splitter_1 which is an instance of the generic submodel flow_splitter

10. To increase the key_component h2o content of the output stream s5 of the flow_splitter flow_splitter_1, call the

Continue?
(Y or N or C)[Y]

appropriate routine attached to the key_component_increase property of the flow_splitter flow_splitter_1

11. This rule group contains the rules and/or rule groups for the submodel flow_splitter

This rule group is selected when the goal_stream s2 is an outlet of the submodel flow_splitter_1 which is an instance of the generic submodel flow_splitter

12. To increase the key_component h2o content of the output stream s2 of the flow_splitter flow_splitter_1, call the appropriate routine attached to the key_component_increase property

Continue?
(Y or N or C)[Y]

```
   of the flow_splitter flow_splitter_1

 The explanation of the all previous rules is complete

** ENTER A CARRIAGE RETURN (<CR>) TO CONTINUE **  [<CR>]

IPSE analysis for this goal ends

  Should I write the knowledge base to a file ?

 yes
 no

Make an appropriate choice
<n>

entering in exergy

 Do you wish to perform exergy analysis

  yes
  no
 help

   <n>
```

END

DATE
FILMED
12/17/92