IDENTIFICATION OF HAZARDS IN CHEMICAL PROCESS SYSTEMS

by

CHRISTOPHER JOHN NAGEL

SUBMITTED TO THE DEPARTMENT OF CHEMICAL ENGINEERING IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF SCIENCE

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

May 24, 1991

Signature of Author_____
Department of Chemical Engineering
May 24, 1991

Certified by_____
George Stephanopoulos
Professor, Chemical Engineering
Thesis Supervisor

Accepted by_____
William M. Deen
Chairman of the Committee for Graduate Students

IDENTIFICATION OF HAZARDS IN CHEMICAL PROCESS SYSTEMS

by

CHRISTOPHER JOHN NAGEL

Submitted to the Department of Chemical Engineering
on May 24, 1991 in partial fulfillment of the
requirements for the Degree of Doctor of Science
in Chemical Engineering

ABSTRACT

Safety of chemical processing plants is a key design factor in the successful operation of processing systems. Successful design technology relies on the ability to identify and eliminate inherent design weaknesses before they are incorporated into the final design. Complete identification and accurate assessment of the hazard potential in the early design stages is particularly important. Changes can then be smoothly integrated into future designs without adversely affecting processing and control complexity or capital and operational costs.

No single methodology exists to unify the systematic study of design alternatives at all stages of the design process; rather, multiple methods are used throughout the design process (development, construction, operation). These methods are self standing, exhibiting a strength at a particular design phase, and are an approximation to hazard analysis and assessment. They cannot guarantee the identification of potential hazards or the underlying root causes leading to them. These synthetic stages, the identification of top level events and the elucidation of underlying root causes, that place the heaviest burden on the evaluation team and determines the quality of the evaluation effort.

An intensive process based methodology has been developed to solve the problem of identifying inherent design weaknesses. The approach begins with the inductive determination of hazardous chemical or physical reactions, or both, and proceeds deductively through the network of processing steps to identify all the root causes initiating a chain of events leading to hazards. The central idea is to begin with potential process interactions and *inductively* identify all feasible chemical and physical reactions. Once feasible reactions become known and their promoters identified the chain of events leading to these reactions are *deductively* identified.

Deductive identification is accomplished by constructing the topological map that satisfied the top level event in the context of the process network. Root causes and their preconditions are then identified and associated with the top level event (the potential hazard) by tracing the influence paths of the variables that define the state of the top level event. The deductive process beginning at the potential reaction set comes with a

guarantee: the complete set of chains of events leading to hazards will be covered by the deductions. Hence, the methodology is complete within the domain of knowledge that identified feasible reactions and the modeling scope which establishes the influence of various variable quantities.

The combined (inductive/deductive) methodological approach lifts the burden of synthetic activity from the hazard evaluation team. An algorithm is proposed for converting topological maps, which establish the chains of events leading to top level events, into fault trees. The methodology and select components which comprise it are illustrated through several case studies. The methodology is illustrated using case studies.

Thesis Supervisor: Dr. George Stephanopoulos, Professor of Chemical Engineering

ACKNOWLEDGEMENTS

I would like to thank my advisor, Professor George Stephanopoulos, for his invaluable technical contributions and the independence he allowed me to have in doing this research. He, probably more than any other single individual, has taught me about the true scope of systems engineering. I am also extremely indebted to Professor Robert Bach: a dear friend whose continued support and fundamental understanding of the behavior of chemical species played an important role in the development of the language for chemical reasoning.

I am also indebted to the members of my thesis committee, Professors F. D. Greene, A. F. Sarofim, J. W. Tester, and P. S. Virk; and to Mr. R. J. Gardner. Without the support of these individuals, the systems approach taken could not have been enabled.

My coworkers, Theodoros Kritikos, Matthew Realff and Jim Johnston also contributed significantly to this research. Matthew Realff, in particular, through his teachings on completeness and his assistance with the proofs contained in Chapter 3; Jim Johnston by his untiring discussions concerning control objectives; and, Theodoros Kritikos through his endless discussions on object-oriented programming

Many other friends have made my years here at MIT much happier. They include from MIT: Dr. Jeffery Kane, Kevin Sparks, Ian Yates, John Carrier, and Bhavik Bakshi - an individual that continually reminds me of the beauty outside of science.

I would like to extend my love and appreciation to my parents and brothers and sisters. They are, and have been, a source of continual inspiration and deserve a great deal of credit for everything that I have or will achieve.

Finally, and most importantly, I would like to thank my best friend, companion, and spouse; the most significant person in my life: Katherine Gundersen. Her belief and undying support is the single reason this trek was possible.

TABLE OF CONTENTS

To

Katherine

and those who support the human condition

There is no virtue in not knowing

what can be known.

- George Bernard Shaw

Chapter 1
Statement of Problem

1.1. Introduction

The safety of chemical processing plants is paramount to their successful operation. Quality design technology relies on the ability to identify and eliminate inherent design weaknesses a priori. Complete identification and accurate assessment of the hazard potential in the early design stages is particularly important, so that changes can be smoothly integrated into future versions (design) without adversely effecting processing and control complexity or capital and operational costs.

1.2. Description

When a process is transferred from the laboratory, a variety of hazards may appear which had earlier been well controlled by the relatively small scale of the discovery effort. Often, discovery operations cannot be translated directly into larger scale development [Brannegan, 1985]. The goal of process development is to provide an efficient and safe process for manufacture.

Throughout a process's development, two factors which may introduce new hazards must be continually examined: change and scale-up [Brannegan, 1985]. Change complicates hazards evaluation by introducing new components whose hazards may be unknown. Scale-up, particularly initial scale-up, can generate significant potential hazards by esculating effects which were thought to be benign.

Since changes often occur throughout the life of the plant, the need to identify hazards as early as possible in the development stages does not imply that hazard identification ends when the design specifications have been approved. In fact, approval of a design means only: "at the time of the study the study team believes that, provided the plant is constructed and operated in accordance with their recommendations, the plant will be acceptably safe" [Lowe and Solomon, 1983]. The first uncontrolled change during construction, or the first unapproved modification during operation, negates this approval. Consequently, hazard identification is a continuing ingredient of safe operations and should be applied, sometimes in a very simple form, to control any changes from the original intentions of the designers.

Several approaches have been presented in the past decade to systematize hazard analysis and hazard identification, but procedural robustness is often constrained by the quality of information available and the expertise of the individuals involved. The more formal methods, such as hazop, FMECA (failure modes, effects, and criticality analysis), fault trees, event trees, and cause-consequence analysis, require information which is often sparse in the early design stages and the certainty of which may be indeterminate. These procedures generate solutions which increase plant complexity and operational costs. They are referred to as extrinsic methodologies. The alternatives for early design analysis are somewhat ad-hoc, being semi-qualitative in nature, and often manifest themselves in the form of process rankings, what-ifs, and, cause and effect analysis.

An alternative approach to loss prevention is to avoid hazards, rather than control them. This approach is feedforward in spirit, evolving the design technology towards inherently safe configurations through the use of codes, guidelines, and checklists. Methodologies which incorporate this strategy are referred to as intrinsic. Although the intrinsic approach can be effective in conceptual/developmental stages of design, it lacks formality and affords no means for complete and consistent hazard identification. In fact, neither of the two approaches guarantee the identification of potential hazards.

1.3. Process trends

The desire to improve the efficiency of the process has led to numerous modifications of the initial process design, all of which may add to the complexity and hence to the development of potential hazards of the process. There are almost as many major variations as there are operating companies. Each of these variations has its own peculiar hazards and each operating company has had to develop its own expertise in its own particular version of the process. As organic chemistry moves increasingly in the direction of faster more complex reactions, it becomes essential to recognize hazards early, so that inherently safe processes can be designed to limit the magnitude of potential consequences. In this spirit, how can designs be made more reliable?

1.4. Outstanding problems

The importance of hazard recognition, in advance of an unwanted event, is recognized. However, risk analysis can be no better than the extent to which hazards are recognized in the first place. Furthermore, the analysis is no better than the analyst's understanding of the plant operations.

Decisions about safety are made continuously. These decisions are made in light of all the uncertainties, and are based on the understanding of the characteristics of the facility and the substances involved. Formal analytical processes may or may not be involved in the decision process. Studies recently indicated that design errors[1] were rarely revealed before startup and accounted for 25% of all accidents [Haastrup, 1983]. The Indian experience with design error is closer to 40% and it has been suggested that if the definition is broadened to include management and organizational aspects of the system, design errors would account for nearly 90% [Batstone, 1987] of all accidents.

To adapt Tolstoy's remark about happy and unhappy families, safe reliable designs are all alike, but every unreliable design creates hazardous situations in its own way. What is needed is a formal, unified method for systematically studying design alternatives at all stages of the design process.

1.5 Significance of Research

The approach developed for automatic hazard analysis is opposite that traditionally used. Beginning with the process, not the equipment. The identification of hazards is inductive and the identification of pathways of precursor events leading to potential hazards is deductive. This approach guarantees completeness within a domain of knowledge.

Predicated on the two postulates that comprise the foundation of this work this approach allows the:

- formalization of hazard identification.
- systemization of hazard analysis.
- complete identification of hazards within a domain of knowledge.

The approach is evolutionary. It is based upon chemistry and engineering science rather than collective team judgment. As such, quantification can be used at any stage in the design process to discriminate among design improvements. More importantly, the use of available knowledge is maximized at each phase in the design sequence. This affords the earliest possible identification of hazards.

---

[1] A design error is deemed to have been committed when the design is changed after an incident.

1.5.1. Goals and contributions

The implementation of this framework constitutes a major advance in the *a priori* identification of hazards. Successful demonstration of this methodology has resulted in the following contributions:

- formal systematization of hazard analysis;
- automation of hazard analysis;
- development of an expressive modeling language for chemical species representation and generation;
- automatic generation of primary, neutral, free radical based products within a homologous series;
- formalization of hazard analysis.

1.5.2. Objectives

The three broad objectives of this research were:

- To develop a formal methodology for the identification of hazards;
- Unification of the identification of hazards through all stages of the design process;
- Completeness of the methodology within the scope of the modeling efforts;

Table 1-1 divides these objectives into 9 specific research goals. Each of these goals addresses fundamental and applied issues of interest in the identification of hazards in chemical process systems.

TABLE 1-1

SPECIFIC RESEARCH GOALS

Goal	Thesis Chapter
Developed a formal methodology which unifies the identification of hazards throughout stages of the design process.	3
Verified the methodology is efficient and complete.	3
Developed a specialized modeling language for reasoning about chemical structures and their generation (CRL) to support the automation of this methodology.	5, 6
Established a formal strategy for the integration of safety into design technology.	4
Validated a metric for discriminating between design alternatives with respect to disturbance mitigation.	4
Determined a basis for optimizing a design technology with respect to disturbance mitigation.	4
Established a means for automatically constructing the network of pathways leading to a hazardous state.	3, 4
Developed a means for creating fault trees from the information associated with the hazardous state and the pathway leading to it.	4

1.6 Bibliography

Batstone R., International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 5.126

Brannegan, D.P., "Hazards Evaluation in Process Development", Chemical Process Hazard Review, American Chemical Society, 1985, p. 18

Haastrup, P., "Design Error in the Chemical Industry", IChemE Symposium Series, No. 80, 1983, p. J15

Lowe, D.R., and C.H. Solomon, "Hazard Identification Procedures", IChemE Symposium Series, No. 80, 1983, p. G8

Chapter 2
Survey of Background and Literature

2.1. Introduction

Hazard identification has always been an integral part of design and operational practice. However, it still is to a large degree an informal process dependent on the experience of those directly involved.

The basic objective of all risk assessment techniques is to examine an actual or proposed installation to identify and assess potentially hazardous situations, their possible consequences and associated risk, in order to provide a rational basis for determining where risk reduction measures are needed. The guiding principle in all hazard analyses is to carry out each step of the analysis in the project stage where it is still possible to make the particular types of changes that the analysis may suggest. Additionally, since any change to reduce the risk of one hazard may actually increase the risk from another hazard or introduce a new hazard, interactions must be considered before the hazard evaluation is terminated.

The intent of the analysis is to provide the design engineer with basic information about safety aspects of the process as well as indicate the reliability of controlling or containing the process. These analyses must be treated with caution however, because of their dependence on assumptions, data bases, and models of varying quality.

2.2. Hazard analysis

Hazard analysis is defined as the systematic investigation of inherent, acute hazards of a process, under normal operating conditions as well as under reasonably foreseeable abnormal conditions. In the analysis, the process is studied apart from the containment and the lay-out of the installation. In order to identify the inherent hazards, the properties of all chemicals involved as well as the characteristics of the process must be studied. The objective of the analysis is to determine the safe limits of the process parameters and to appreciate the effects when the process parameters move outside these limits.

The overall goal of a typical hazards analysis of a chemical process is to develop operating and design criteria intended to prevent or mitigate the effects of identified

hazards. In order to accomplish this goal, the analysis must identify critical material characteristics and operating conditions which provide a realistic means of controlling the likelihood or severity of the various types of hazards. To be timely and cost effective, any testing program conducted in support of the hazards analysis effort should focus on the identified control parameters. Typically, a process review would include a review of: the process chemistry (including principal and expected side reactions), system thermodynamics, heat generation versus heat removal capability of the system, reactor and storage tank vent sizing, reactive chemicals test data, planned operation, consequences of inadequate procedures, start-up and shutdown procedures, and the lines of defense employed to avoid reactive chemical accidents at each point. This does not guarantee that the project will be free of reactive chemical problems, but it does reduce the chance that something will be overlooked.

Structured hazard identification methods can be roughly classified in two groups [Boykin and Kazarians, 1987, and, Ozog and Bendixen, 1987]

1. Comparative methods which rely upon comparing the design with some recognized code or standard.

2. Fundamental methods which can be applied in almost any situation.

Individual experience, broadened by information from the experience of others, is the fundamental ingredient of hazard identification. However, hazard control techniques require that individual experiences be collected and recorded in a form which makes the knowledge readily accessible to the people developing and designing the equipment. National and international codes and standards are examples of how this is done. The authors of a code are implying that, "based on their collective experience, equipment designed to their code will be acceptably safe" [Brannegan, 1985]. Consequently, codes and practices provide minimum standards (norms) against which deviations from safe practices can be identified and appraised. Systematic comparison of a design specification with recognized codes and practices forms the basis of one family of hazard identification methods which can be described as *comparative methods*. An important advantage of these methods is that the lessons learned through many years of experience are incorporated in the company's practices and thus are available to be used at all stages in the design and construction of the plant. The main task of the hazard identification study is to ensure that the company's practices, and therefore its past experience, have indeed been incorporated in the design.

In the case of new processes, which are outside the scope of existing codes of practice, prior experience will not be available. The knowledge which can be used in the hazard identification procedure is now strongly dependent upon information obtained a priori. This information is derived from the efforts of the research and development engineers together with the personal knowledge of the hazard identification team members. Consequently, the hazard identification methods used in such cases have to be directed towards stimulating the team members to use their own experience of safe and unsafe as the standard against which to appraise the design. These techniques, which are referred to as *fundamental methods*, are essentially structured ways for stimulating a group of people to apply their knowledge to the task of identifying hazards mainly by raising a series of "what if" questions. Fundamental methods of hazard identification are aimed at two particular outcomes. Firstly, there is the identification of serious incidents which may result directly in danger to employees or the public, or in financial loss. These are usually known as the "top events". Secondly, the fundamental methods can be used to identify the underlying root causes which can lead to the top events.

2.3. Predictive hazard analysis

Various procedures and techniques (Figure 2-1.) have been developed to aid identification of hazards throughout the various stages of new projects as well as in existing operating units.

- o Checklists
- o Safety Review
- o Siting
- o Relative Ranking
- o Preliminary Hazard Analysis
- o "What If" Analysis
- o Hazard and Operability Analysis
- o Failure Modes, Effects, and Critical Analysis
- o Fault Tree Analysis
- o Event Tree Analysis
- o Cause – Consequence Analysis
- o Human Error Analysis

Figure 2-1. State of the Art

When properly applied, these various procedures and techniques can aid hazard identification (Figure 2-2.). However, they will not work well if only followed mechanically without real thought on the part of the reviewers. The order of steps in a typical hazard evaluation is as follows [Battelle, 1985]:

1. Identify the hazards inherent in the process/plant.
2. Estimate the consequences that could result from the hazards identified in Step 1.
3. Identify opportunities to reduce the consequences in Step 2.
4. Identify initiating events of accidents that could lead to the consequences estimated in Step 2.
5. Estimate the probabilities of the initiating events.
6. Identify opportunities to reduce the probabilities of the initiating events.
7. Identify the event sequences of accidents (system responses) that could lead to the consequences estimated in Step 2.
8. Estimate the probabilities and consequences of the accident event sequences identified in Step 7.
9. Identify opportunities to reduce the probabilities and/or the consequences of the accident event sequences identified in Step 7.
10. If necessary, do quantitative hazard evaluations to reduce the uncertainty in the estimates of probabilities and consequences and identify optimum investments for obtaining a level of risk deemed acceptable.

Actions to reduce risk are generally of five kinds [Battelle, 1985]:

1. A change in the physical design or control system.
2. A change in operating method.
3. A change in process (pressure, temperature).
4. A change in the process materials.
5. A change in the test and inspection/calibration of key safety items.

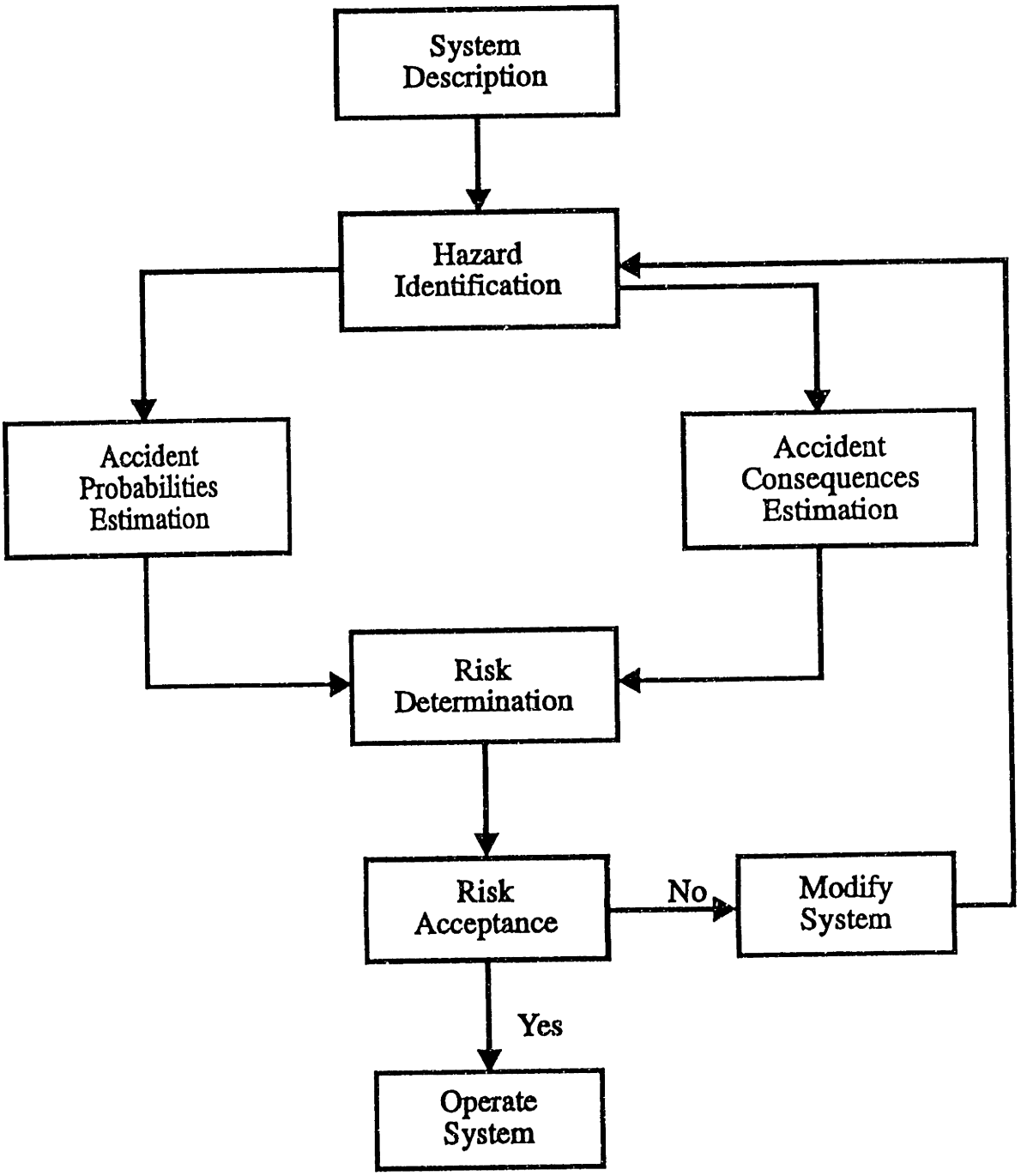


Figure 2-2: Predictive Hazard Evaluation

These actions may be categorized as follows [Battelle, 1985]:

1. Actions which eliminate the hazard.
2. Actions which eliminate or reduce the consequences.
3. Actions which reduce the likelihood to an acceptable level.

In general, methods which identify actions that eliminate, avoid, or reduce potential hazard of a particular design are commonly referred to as intrinsic. The ensuing design concepts are inherently safe utilizing (passive) mitigating features which do not require any action (either manual or automatic) or utilities. Such features might include: change chemistry, change processing technology, reduce inventories of materials, appropriate spacing of equipment, or containment barriers. Alternatively, methods which identify actions that reduce the likelihood of a hazard through control are referred to as extrinsic in spirit. Emerging designs utilize (active) mitigating systems which do require some action to be effective. Examples might include relief valves, detectors, and fire fighting provisions.

The techniques used to exploit the methodologies associated with predictive hazard analysis can be further distinguished by the type and quality of knowledge involved. Essentially, the techniques used to assess damage levels are primarily quantitative in nature while those used to identify and assign risk to a troublesome area are primarily qualitative. The methods marshaled to advance predictive hazard analysis are described below.

2.3.1. Quantitative methods

Microscopic:

Quantitative methods [Boykin and Kazarians, 1987, Ozog and Bendixen, 1987, and Slater, Corran, and Pitblado, 1987] are utilized on both microscopic and macroscopic levels. Microscopic quantification often manifests itself as an experimental testing method [Hoffmann, 1985] (see Table 2-1) which detects various reactive chemical problems.

Table 2-1. Experimental Testing Methods	
DTA	Differential Thermal Analysis
DSC	Differential Scanning Calorimetry
TGA	Thermo Gravimetric Analysis
ARC	Accelerating Rate Calorimetry
BSC	Bench Scale, Heat Flow Calorimetry
SEDEX	Sensitive Detector of Exothermic Processes
Others	Oven Tests, Dewar Tests, Hot Plate Tests, etc.

Among the purpose which analytical tests serve are the determination of the onset of exothermic (endothermic) decomposition. It is important to note that these tests are accurate only for the exact conditions under which they were run. Actual conditions in a process may require tests specially adapted for the system.

Chemicals participating in the process can be hazardous depending on their properties and the process conditions. Physical, chemical and toxicological properties of all substances have to be considered. Process hazards may be caused directly, by the release of a chemical in the atmosphere, or indirectly by introducing a hazardous situation because of the instability or reactivity of the chemicals involved. Therefore, not only are data about the flammability, explosivity, and toxicity relevant, but also aspects like the disintegration and polymerization of the chemicals due to contact with certain construction materials, light, storage time, and temperature.

Macroscopic:

Using the information derived from micro-quantitative testing and various information derived from environmental and structural influences, macroscopic techniques [Boykin and Kazarians, 1987, Ozog, and Bendixen, 1987, and Slater, Corran, and Pitblado, 1987] use effect models to assess damage to a structure or a community, although, the vulnerability of the environment is not taken into account. Several effect models are listed below:

1. Discharge

   Discharge of liquids, gases and vapors

   Two-phase discharge

2. Evaporation

   Evaporation of liquids on land

Evaporation of liquids on water

3. Dispersion

   Gaussian dispersion models

   Heavy gas dispersion models

4. Heat radiation

5. Unconfined vapor cloud explosions

   Explosion models

Parameters accounted for often include: properties of the released fluid, release conditions, weather conditions, and local topography (plot layout and ground roughness).

2.3.2. Qualitative method

[Boykin and Kazarians, 1987, Ozog and Bendixen, 1987, Battelle,1985, Cox, 1987, Atallah, 1980, Culbertson and Searson, 1983, Ozog, 1987]

2.3.2.1. Intrinsic

Checklists:

Checklists manifest themselves as experienced based questionnaires and, as such, are limited to the experience base of the author(s). They are engineered to demonstrate design compliance with standard procedures and provide direction for standard evaluation of chemical plant hazards. Checklists can be as detailed as necessary to satisfy the specific situation, but should be applied conscientiously in order to identify problems that require attention and to ensure that standard procedures are being followed.

A checklist is easy to use and can be applied to each stage of a project or plant development. A checklist is a convenient means of communicating the minimal acceptable level of hazard evaluation that is required for any job, regardless of scope.

Safety Reviews:

Safety reviews can vary from an informal walk through on-site inspection that is principally visual, with emphasis on housekeeping, to a formal week long examination by a team with appropriate backgrounds and responsibilities. Such a program is intended to identify plant conditions or operating procedures that could lead to an accident and significant losses in life or property.

Hazard Indices:

Hazard indices such as that developed by Dow Chemical Company [Fire and Explosion Index, 1976] and extended by Lewis [Lewis,1979], are methods which are designed to give a quantitative indication of the potential for hazardous incidents associated with a given plant.design. The methods assign penalties and credits based on plant features. Penalties are assigned to process materials and conditions that can contribute to an accident. Credits are assigned to plant safety features that can mitigate the effects of an accident. These penalties and credits are combined to derive an index that is a relative ranking of the plant risk. Estimates of consequences in terms of cost and outage time can also be included in the evaluations. These methods are particularly useful in the early stages of hazard assessment in that they require a minimum of process and design data and can graphically demonstrate which areas within the plant require more detailed attention. They can also help to identify which of several competing process routes will contain the least inherent hazards.

The primary difference between the Dow Index and the Mond Index is that the latter specifically addresses material toxicity in addition to flammability and reactivity in assigning material factors to the process unit. The Dow Index may actually be easier to use because of the extensive use of tables and graphs in place of traditional equations, but both methods use the same basic calculation method.

Preliminary Hazard Analysis (PHA):

The main purpose of this analysis is to recognize hazards early. This is achieved by speculating about possible causes, consequences, and corrective measures early in the design process. It is generally applied during the concept or early development phase of a process plant and can be very useful in site selection. PHA is a precursor to further hazard analyses.

2.3.2.2. Extrinsic

"What If":

The purpose of a "What If" analysis is to consider carefully the result of unexpected events that would produce an adverse consequence. The method involves examination of possible deviations from the design, construction, modification, or operating intent. The questions are divided into specific areas of investigation (usually related to consequences

of concern), such as electrical safety, fire protection, or personnel safety. It requires a basic understanding of what is intended and the ability to mentally combine or synthesize possible deviations from design intent that would cause an undesired result.

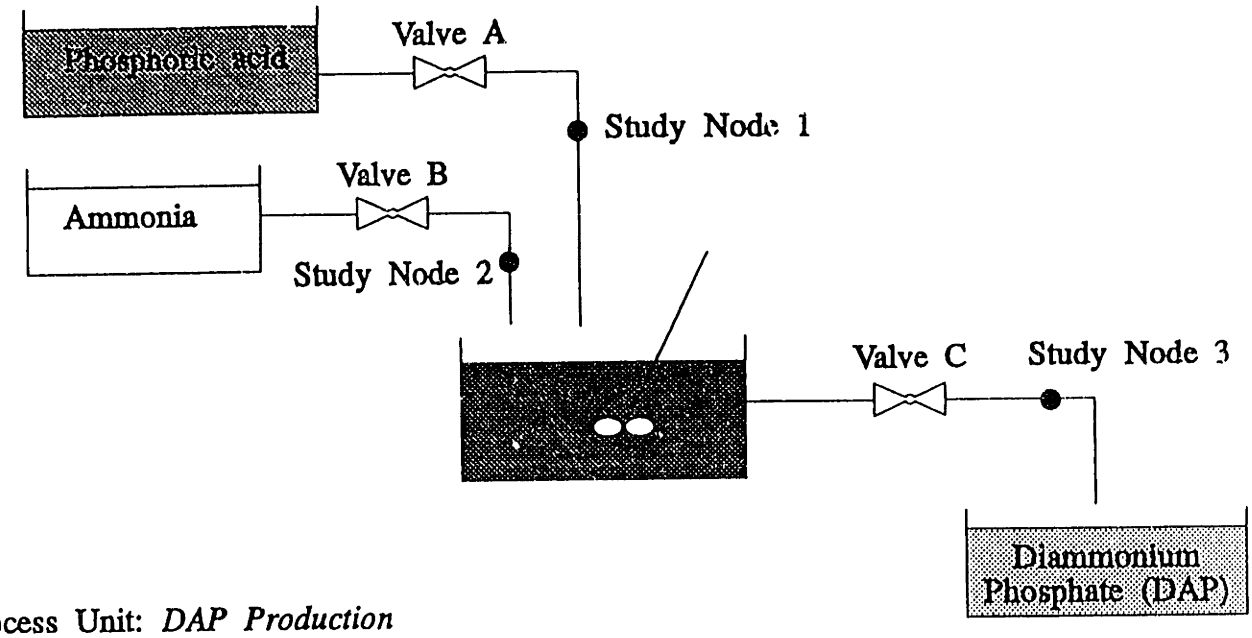
Hazard and Operability Studies (HazOp):

The HazOp study was developed to identify hazards in a process plant and to identify operability problems which, though not hazardous, could compromise the plant's ability to achieve design productivity. Thus, a HazOp goes beyond hazard identification. Although originally developed to anticipate hazards and operability problems for new and/or novel technology where past experience is limited, it has been found to be very effective for use at any stage in a plant's life from final design on. In addition, one variation of the HazOp has been developed specifically to address preliminary design.

The approach taken is to form a multidisciplinary team that works together (brainstorms) to identify hazards and operability problems by searching for deviations from design intents. An experienced team leader systematically guides the team through the plant design using a fixed set of words, called guide words, or uses checklists. These guide words are applied at specific points or study nodes in the plant design to identify potential deviations of the plant process parameters at those nodes. The nodes are usually specified by the team leader before the meetings. For example, the guide word "no" combined with the process parameter "flow" results in the deviation "no flow". The team then agrees on possible causes of the deviations (e.g., operator error shuts off pumps) and the consequences (e.g., product contamination). If the causes and consequences are realistic and significant, they are recorded for follow-up action which takes place outside of the study. In some cases, the team identifies a deviation with a realistic cause but of unknown consequence (e.g., unknown reaction product) and recommends studies to determine the possible consequences. While this method can be used without direct reference to engineering standards it requires a broad documentation of the points studied to demonstrate the quality of the study (Figure 2-3). This aspect must be very carefully controlled.

Failure Modes, Effects, and Criticality Analysis (FMECA):

FMECA is a tabulation of the system/plant equipment, their failure modes, each failure mode's effect on the system/plant, and a criticality ranking for each failure mode. [A Failure Modes and Effects Analysis (FMEA) is equivalent to a FMECA without a



Process Unit: *DAP Production*

Node: 1

Process Parameter: *Flow*

Guide Word	Deviation	Consequences	Causes	Suggested Action
No	No Flow	Excess ammonia in reactor Release to work area	1. Valve A fails closed 2. Phosphoric acid supply exhausted 3. Plug in pipe pipe ruptures	Automatic closure of valve B on loss of flow from phosphoric acid supply.
Less	Less Flow	Excess ammonia in reactor Release to work area, with amount released related to quantitative reduction in supply. Team member to calculate toxicity vs. flow reduction.	1. Valve A partially closed 2. Partial plug or leak in pipe	Automatic closure of valve B on reduced flow from phosphoric acid supply. Set point determined by toxicity vs. flow calculation.
More	More Flow	Excess phosphoric acid degrades product. No hazard to work area.		
Part of	Normal flow of decreased concentration of phosphoric acid	Excess ammonia in reactor. Release to work area, with amount released related to quantitative reduction in supply.	1. Vender delivers wrong material or concentration	Check phosphoric acid supply tank concentration after charging.

Figure 2-3: HazOp Analysis

criticality ranking.] The failure mode is a description of how equipment fails (open, closed, on, off, leaks, etc.). The effect of the failure mode is the system response or accident resulting from the equipment failure. FMECA identifies single failure modes that either directly result in or contribute significantly to an important accident. Human/operator errors are generally not examined by an FMECA; however, the effects of a misoperation are usually described by an equipment failure mode. FMECA is not efficient for identifying combinations of equipment failures that lead to accidents.

The method is especially useful for the analysis of very critical processes but is extremely time consuming if applied on too broad a scale. The FMECA can be performed by two analysts or a multidisciplinary team of professionals.

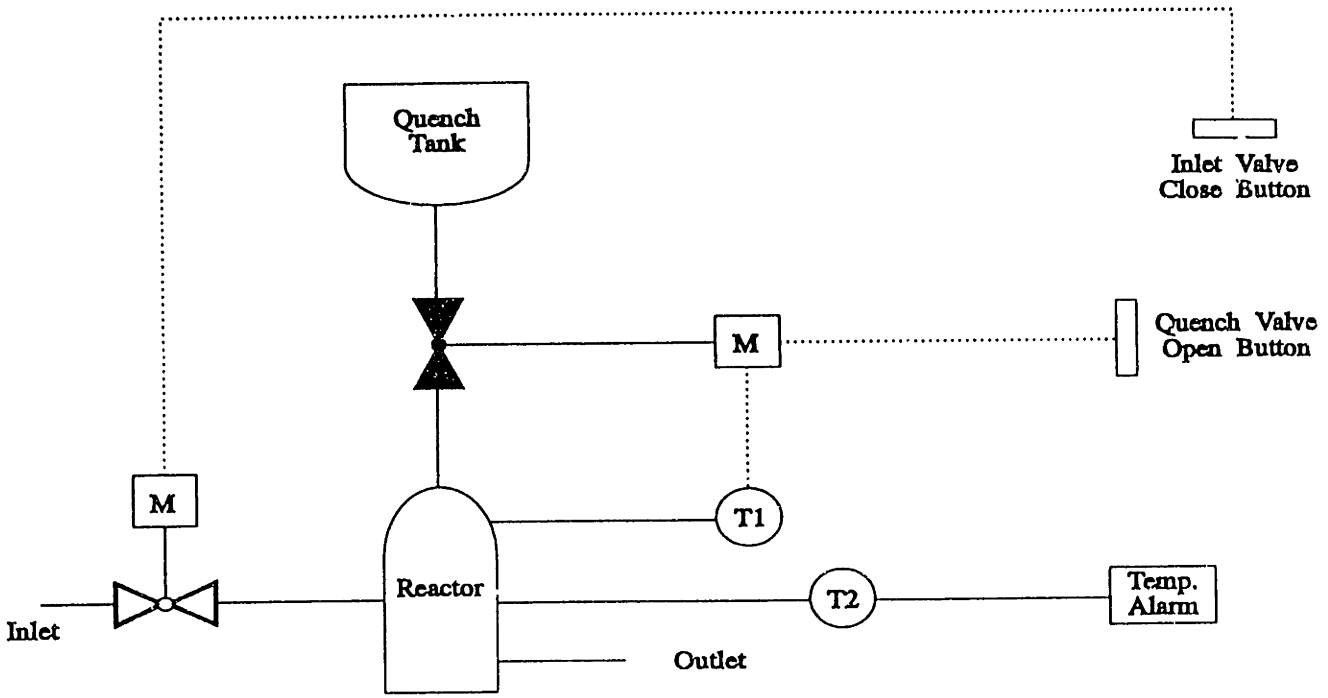
Fault Tree Analysis (FTA):

Fault Tree Analysis is a deductive technique that focuses on one particular event and provides a method for determining causes of that accident event. The fault tree itself is a graphic model that displays the various combinations of equipment faults and failures that can result in the accident event (Figure 2-4.). The solution of the fault tree is a list of the sets of equipment failures that are sufficient to result in the accident event of interest. FTA can include contributing human/operator errors as well as equipment failures.

Ranking the minimal cut sets is the final step of the fault tree analysis procedure. Structural importance is reflected by the number of basic events that are in each minimal cut set. In this type of ranking, a one event minimal cut set is more important than a two event minimal cut set; a two event set is more important than a three event set; and so on. This ranking implies that one event is more likely to occur than two events, two events are more likely to occur than three events, etc.

The strength of FTA as a qualitative tool is its ability to break down an accident into basic equipment failures and human errors. This allows the safety analyst to focus preventive measures on these basic causes to reduce the probability of an accident.

Both failure modes and effects analysis and fault tree analysis are useful aids to hazard identification as they both structure and document the analysis. However, because they involve very detailed analysis of components and operations, their use in the process industry is mainly limited to identification of special hazards where they form the basis of quantification of risks.



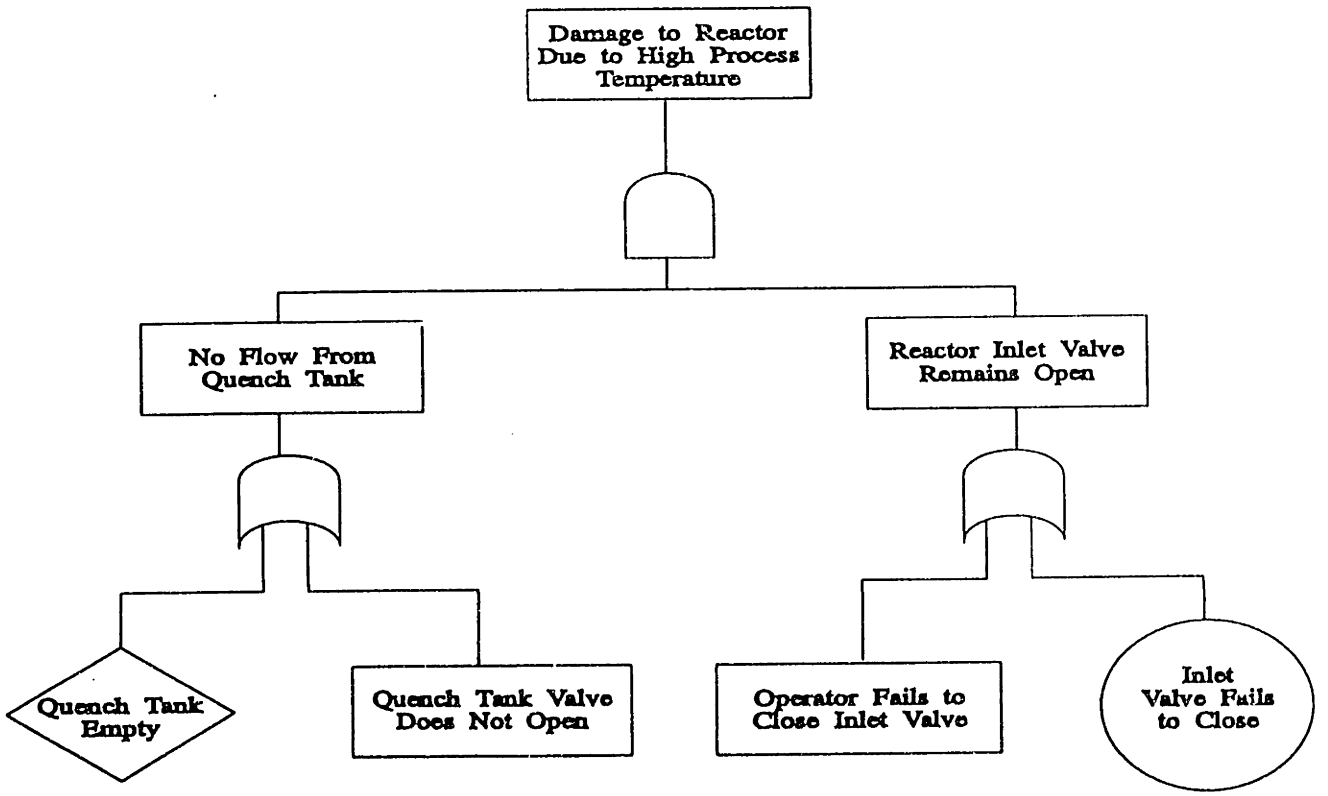


Figure 2-4: Fault Tree Analysis

Event Tree Analysis (ETA):

Event trees are a modified form of the decision trees traditionally used in business applications. Event trees provide a precise way of recording the accident sequences and defining the relationships between the initiating events and the subsequent events that combine to result in an accident. Then by ranking the accidents, or through a subsequent quantitative evaluation, the most important accidents are identified. Event trees are well suited for analyzing initiating events that could result in a variety of effects. An event tree emphasizes the initial cause and works from the initiating event to the final effects of the event. Each branch of the event tree represents a separate effect (event sequence) that is a clearly defined set of functional relationships.

Event tree analysis is a technique for evaluating potential accident outcomes resulting from a specific system failure or human error known as an initiating event. Event tree analysis considers operator response or safety system response to the initiating event in determining the potential accident outcomes. The results of the event tree analysis are accident sequences; that is, a chronological set of failures or errors that define an accident. These results describe the possible accident outcomes in terms of the sequence of events (successes or failures of safety functions) that follow an initiating event. Event tree analysis is well suited for systems that have safety systems or emergency procedures in place to respond to specific initiating events.

Event tree analysis evaluates potential accident outcomes that might result following an equipment failure or process upset known as an initiating event. Unlike fault tree analysis, event tree analysis is a forward thinking process; that is, the analyst begins with an initiating event and develops the following sequences of events that describe potential accidents, accounting for both the successes and the failures of the safety functions as the accident progresses.

Cause-Consequence Analysis (CCA):

Cause-consequence analysis combines the forward thinking features of event tree analysis with the reverse thinking features of fault tree analysis. The result is a technique that relates specific accident consequences to their many possible basic causes. Its advantage to the analyst is toward the consequences of the event and backward toward the basic causes of the event.

The solution of the cause-consequence diagram for a particular accident sequence is a list of accident sequence minimal cut sets. These sets are analogous to fault tree minimal cut sets because they represent all the combinations of basic causes that can result in the accident sequence. A quantitative analysis using these sets can provide estimates of the frequency of occurrence of each accident event sequence.

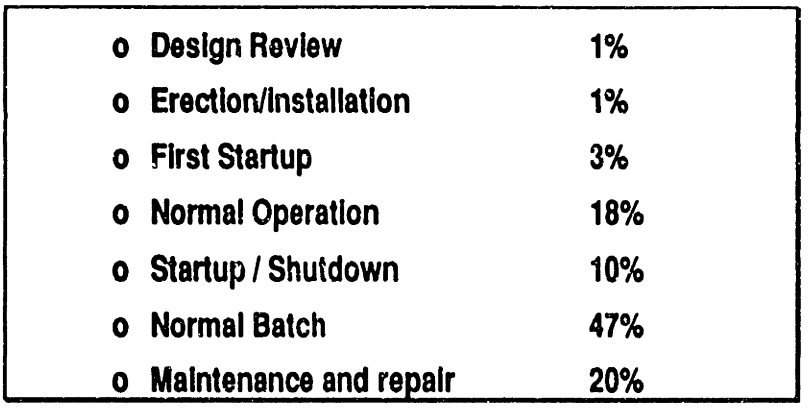
A major strength of cause-consequence analysis is its use as a communication tool: the cause-consequence diagram displays the interrelationships between the accident outcomes (consequences) and their basic causes. The method can be used to quantify the expected frequency of occurrence of the consequences if the appropriate data are available.

Human Error Analysis (HEA):

Human error analysis is a systematic evaluation of the factors that influence the performance of human operators, maintenance staff, technicians, and other plant personnel in the plant. Its primary purpose is to identify potential human errors and their effects or identify the cause of observed human errors.

2.4. Critique

Lees (1980) stated that "The safety of the plant is determined primarily by the quality of the basic design concept rather than by the addition of special safety features." This point is difficult to over emphasize. The degree to which it is economic to eliminate (as opposed to control) a hazard is very much dependent upon when the hazard is first recognized. By the time the design has reached the stage of being sufficiently well documented to allow a detailed hazard identification study to be done, the flexibility to eliminate hazards, entirely, is very much reduced (see Figure 2-5). If the time of hazard/error detection is to be shifted to predesign review, a less encumbered approach must be developed and integrated into the design process.



Time of Error Detection

[No. of cases = 215]

Figure 2-5. Design Analysis

Hazard evaluation should be a continuing process from the conception of the processing scheme to plant shutdown and decommissioning. However, because conventional methods are limited by their scope and useful life span, integrated evaluation into the design process is difficult to achieve. More importantly, all conventional methods are incomplete. Their ineffectiveness is inherent to their approach and their lack of expressive power. This results in two principal deficiencies:

1. Strength of analysis is dependent upon the identification of hazards and/or events leading to hazards.

2. Inability to estimate design quality.

These weaknesses prevent an adequate balance of:

1. early identification to avoid costly redesign or construction modifications,

2. postponement of evaluation to await more detail,

3. avoidance of costly duplication of effort,

from being achieved. Currently, there is no single solution to the problem. Rather, multiple methods are used over the design process (development, construction, operation). These methods each suffer from particular deficiencies which are born out from their approach.

Intrinsic methods, for example, although they tend to increase the quality of a design, must generally be accomplished in the early engineering stages. The window of opportunity for this action is very brief since the incorporation of intrinsic safety features at a late stage in the design will usually require major design changes with consequent

cost and schedule impact. Furthermore, although many of the standards and good practices are the result of lessons learned from analysis of accidents and near accidents they do not provide a creative search for new hazards when experience is lacking nor can they provide quantification as to the quality of a particular design.

Heuristics have been also proposed (Table 2-2.) to fortify intrinsic methods [Dale, 1987, and Kletz, 1985]. They are ad hoc, offering no metric of sound origin to discriminate between design concepts. However, they can be suggestive as to a design's quality if they are intelligently applied. Blind usage though, often leads to unforeseen difficulties.

Extrinsic methods attempted to solve the quantification issue by beginning with a detailed design. Unfortunately, by the time a detailed design is available it is often too late to avoid hazards and the only economic alternative is their control. While controlling hazards, in principle leads to an acceptable solution there is no assurance that the envisioned control scheme will be effective in mitigating all eventual outcomes.

Furthermore, while these methods enjoy a more robust foundation they are not firmly established in chemical engineering science but rather in probability theory. Consequently, although some degree of quantification is afforded as to the level of (process) risk there is no measure as to the quality of a particular design concept. This occurs because there is no technique for discriminating between alternative options for improvement; this is still left to the team's collective judgment. Often there is not sufficient information to accurately establish the frequencies of an accident scenario, or there sometimes is not sufficient information about the physical properties of a substance (under unusual conditions) to accurately establish the accident scenarios. This lack of sufficient information introduces a second type of uncertainty, which denotes our (the analysts') level of confidence in or state of knowledge about what the true values of a parameter or the true form of an accident model are. In practice, it is not uncommon to encounter large (or wide) uncertainties[7].

More importantly, the fundamental flaw in any method which is based on controlling hazards lies in the assumption that they have the ability to both identify accurately and pin-point precisely, the location of a future hazard. Virtually all experience suggests that

[7]For example, the frequency estimates for the failure of a pump or a valve may range by two orders of magnitude.

the contrary is true. In fact, where we suspect the possibility of a hazard, we take added precautions to so construct and reinforce so that the hazard will not occur.[8]

In summary, conventional methods are an approximation to hazard analysis and assessment. They are incomplete because their method of deduction and axiom set are not complete. Their performance and effectiveness is fundamentally limited by their lack of expressive power. They avoid model corrections by seeking solutions from techniques which do not have firm chemical engineering science foundations.

Table 2-2. Heuristics
Use Less Hazardous Materials
Reduce Inventories of Hazardous Material
Utilize In-situ Reactions for Limiting Hazardous Intermediates
Minimize Hazardous Process Conditions
Critical Reviews of Design Safety
Build in Process Containment
Minimize the Quantities Released

2.5 Bibliography

Atallah, S., "Assessing and Managing Industrial Risk", Chemical Engineering, September 8, 1980

Battelle, "Guidelines for Hazard Evaluation Procedures", AIChE Press, 1985

Boykin, R.F. and M. Kazarians, "Quantitative Risk Assessment for Chemical Operations", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.87

Brannegan, D.P., "Hazards Evaluation in Process Development", Chemical Process Hazard Review, American Chemical Society, 1985, p. 18

Cox, R.A., "An Overview of Hazard Analysis", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.37

Culbertson, T.L. and A.H. Searson, "Exxon Facility Design Assessment and Control of Hazards", internal publication, September, 1983

[8]If one reviews the past accidents involving manufacturing plants, it is immediately apparent that hazards occurred at locations that would have been virtually impossible to predict - often at points substantially removed from the locations thought to be the primary risk.

Dale, S.E., "Cost Effective Design Considerations for Safer Chemical Plants", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, 1987, AIChE, p. 3.79

Fire and Explosion Index, Corporate Safety and Loss Prevention, Dow Chemical Company, 1976

Hoffmann, J.M., "Chemical Process Hazard Review", Chemical Process Hazard Review, American Chemical Society, 1985, p. 1

Kletz, T.A., "Make Plants inherently Safe", Hydrocarbon Processing, September 1985, p. 172

Lees, F.P., "Hazard Warning Structure: Some Implications and Applications", IChemE Symposium Series No. 80, 1983, p. J1

Lewis, D.J., "The Mond Fire Explosion + Toxicity Index - A Development of the Dow Index", AIChE Loss Prevention Symposium, Houston, Texas, April 1979

Ozog, H. and L.M. Bendixen, "Hazard Identification and Quantification", Chemical Engineering Progress, April, 1987, p. 55

Ozog, H., "Hazard Identification Analysis and Control", Chemical Engineering, February 18, 1987, p. 161

Slater, Corran, and Pitblado, eds., "Major Industrial Hazards project Report", The Warren Centre for Advanced Engineering, University of Sidney, May, 1987

Chapter 3
A Methodology for Automating Hazard Identification and Analysis
Part I: Inductive Identification of the Chemical Basis of Hazards

Summary

No single methodology exists to unify the systematic study of design alternatives at all stages of the design process. Rather, multiple methods are used throughout the design process (development, construction, operation). These methods are self standing, exhibiting a strength at a particular design phase and are an approximation to hazard analysis and assessment. They cannot guarantee the identification of potential hazards or the underlying root causes leading to them. These synthetic stages: the identification of top level events and the elucidation of underlying root causes, place the heaviest burden on the evaluation team and determines the quality of the evaluation effort.

An intensive process based methodology has been developed to solve the problem of identifying inherent design weaknesses. The approach begins with the inductive determination of hazardous chemical or physical reactions or both and proceeds deductively through the network of processing steps to identify all the root causes initiating a chain of events leading to hazards. The central idea is to begin with potential process interactions and *inductively* identify all feasible chemical and physical reactions. Once feasible reactions become known and their promoters identified the chain of events leading to these reactions are *deductively* identified. The deductive process beginning at the potential reaction set comes with a guarantee: the complete set of chains of events leading to hazards will be covered by the deductions. The combined (inductive/deductive) methodological approach lifts the burden of synthetic activity from the hazard evaluation team. The methodology is illustrated using case studies.

3.1. Introduction

The safety of chemical processing plants has become paramount to their successful operation. Quality design technology relies on the ability to identify and eliminate inherent design weaknesses a priori. Complete identification and accurate assessment of the hazard potential in the early design stages, so that changes can be smoothly integrated into future (design) versions without adversely affecting processing and control complexity or capital and operational costs.

Since 1950 there has been an exponential increase in the number of hazardous incidents [Carson and Mumford, 1979]. The increase in incidents can be directly measured in terms of increased fatalities and economic loss. The health hazards associated with these incidents have resulted in: (i) explosions resulting in injury or fatality due to shock blasts, flying projectiles, falls, electrical shock, or impact with stationary objects; (ii) toxic effects on the skin such as acid, caustic, or heat burns, dermatitis or chronic toxic absorption (e.g. HCN); (iii) toxic effects due to inhalation causing acute irritation (e.g. H_2S), acute non-irritation (e.g. CO), chronic irritation (e.g. methyl isocyanate, asbestos), or asphyxiation (e.g vinyl chloride, N_2).

Studies have shown that design errors are rarely revealed before start up; they account for 25% of all accidents [Haastrup, 1983]. The Indian experience with design error is closer to 40%. It has been suggested that if the design error definition is broadened to include management and organizational aspects of the system, design errors would account for nearly 90% [Batstone, 1987] of all accidents. Moreover, the percentage of precursors perceived known at the time of the incident varies depending on the individuals spoken to. Our survey suggests that plant personnel believe 90-100% of precursors leading to a hazard are known at the time of the incident, hazard specialists believe 40-60% of precursors leading to a hazard are known at the time of the incident, and insurance analysts believe 20 to 30 percent of precursors leading to a hazard are known at the time

of the incident. Such data are exacerbating; they suggest, to some, that improved hazard identification methods are unnecessary, possibly even unwanted. Yet, incidents continue.

3.1.1 Hazard Analysis

The basic objective of hazard analysis is to identify and assess potentially hazardous situations, their possible consequences and associated risk, in order to provide a rational basis for determining where risk reduction measures are needed. Since quality design technology relies on the ability to identify and eliminate inherent design weaknesses a priori, the goal is to develop operating and design criteria intended to prevent or mitigate the effects of identified hazards [Lees, 1980]. The evaluation process must be continuous, from the conception of the processing scheme to plant shutdown and decommissioning, if it is to be effective [Lowe and Solomon, 1983].

Two predominate strategies are used in conventional hazard analysis to achieve this goal: avoidance of the effects leading to hazards and control of the events leading to hazards. The (more) formal methods, such as hazop, FMECA (failure modes, effects, and criticality analysis), fault trees, event trees, and cause-consequence analysis, require information which is often sparse in the early design stages and the certainty of which may be indeterminate. These methods tend to control an identified hazard, or the conditions leading to the hazardous state, and often generate solutions which increase plant complexity and operational costs; they are referred to as extrinsic methodologies. The alternative approach to loss prevention strives to avoid hazards and their associated precursor states rather than control them. The approach is feed forward in spirit, evolving the design technology towards inherently safe configurations through the use of codes, guidelines, and checklists. Methodologies using this strategy are called intrinsic methodologies. The approach can be effective in conceptual/developmental stages of design where equipment changes are easily made without adversely affecting

construction costs and schedules (see Figure 3-1), but it lacks formality and affords no means for complete and consistent hazard identification.

Although many approaches employing various strategies have been presented to systematize hazard analysis and hazard identification, the methodologies lack procedural robustness. They are often compromised by the quality of information available or the expertise of the individuals involved or both [Boykin and Kazarians, 1987, Ozog and Bendixen, 1987, Battelle, 1985, Cox, 1987, Atallah, 1980, Culbertson and Searson, 1983, Ozog, 1987, Slater, Corran, and Pitblado, 1987, Mosleh, Bier, and Apostolakis, 1988, Hoffmann, 1985, Dale, 1987]. What is needed is a formal, unified method for systematically, automatically, and completely identifying potential hazards and pathways leading to hazards in design alternatives - at all stages of the design process. But how does one design such a methodology?

3.1.2 Premise of Traditional Approaches

The above weaknesses are all due to a *lack of representational expressiveness.* Conventional methods are an approximation to hazard analysis and assessment because their method of deduction and axiom set are incomplete: a consequence of representational inexpressiveness. This weakness can manifest itself in the following ways: (i) a methodology exhibits strength in a particular design phase; (ii) a methodology is incapable of transferring information derived at one phase to another (nor can it reason about the process and its surrounding environment); (iii) discrimination among designs varies depending upon the technique employed; (iv) complete identification of hazards can not be guaranteed (and therefore the quality of the analysis can not be assessed).

The fundamental premise of traditional approaches is that there is no unifying theme (in hazards analysis); that all hazards are different; that every unreliable design creates

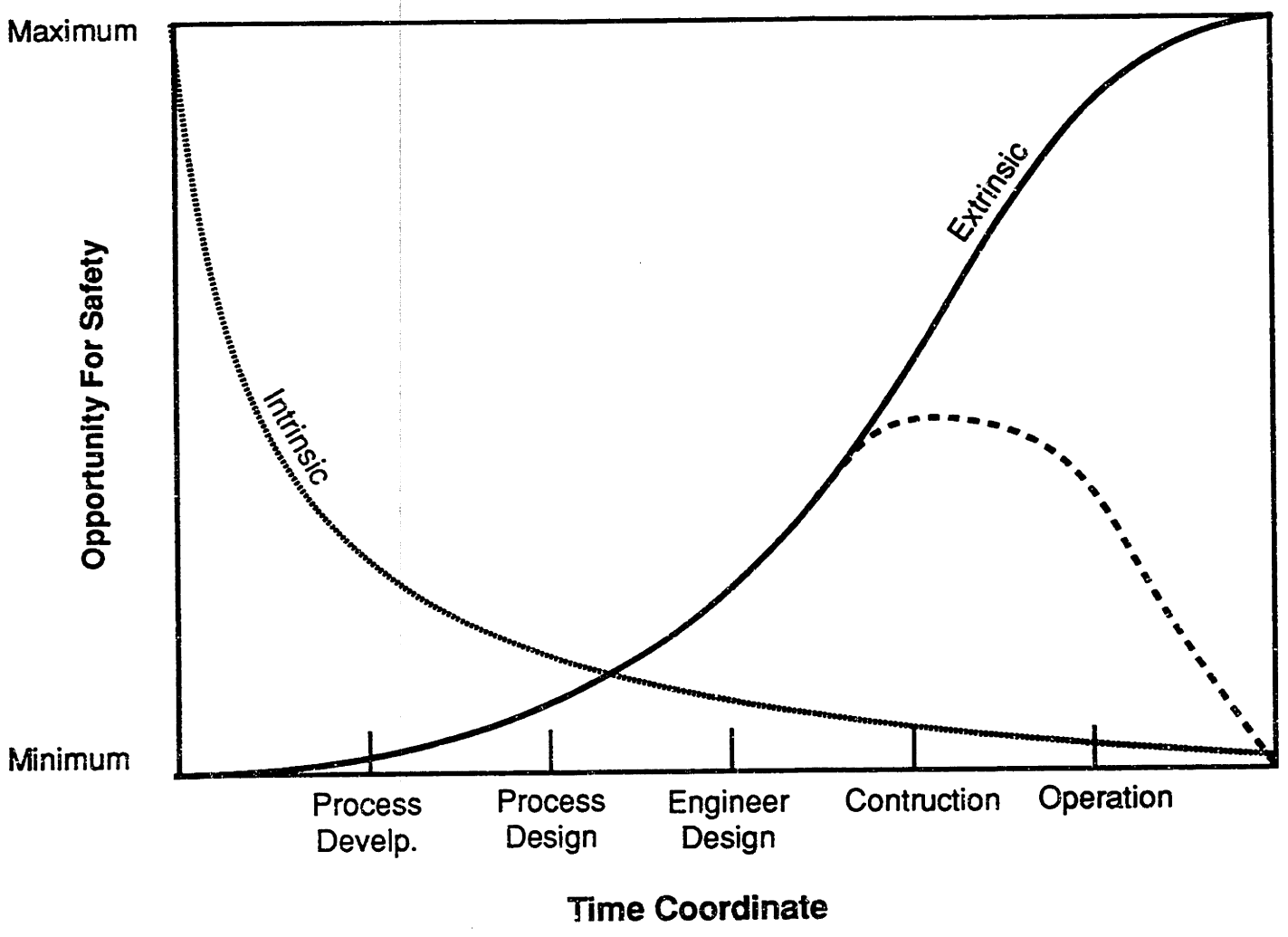


Figure 3-1: Timing of Design Changes

hazardous situations in its own way. The stand alone nature of past approaches is evidence of this statement (see Figure 3-2). We have found the contrary to be true: representational limitations of past efforts have prevented the exploitation of unifying themes.

Moreover, since conventional methodologies do not include *explicitly* information such as the following: (i) Underlying assumptions on operational mode, phase equilibria, fluid mechanical aspects, reaction rates, mechanism of mass and heat transfer or both, and selected approaches in estimating the value of thermodynamic properties; (ii) Simplifications made by the analyst to limit the model's validity over a given range of conditions or to underscore the relative importance of various physicochemical phenomena; (iii) Missing relationships including qualitative relationships, order-of-magnitude reasoning, and inequalities; (iv) Scope of the task, that is, what was the process intended for.

Efforts in engineering science reinforce these observations [Stephanopoulos, 1987]. Many tasks are not achievable if representational expressivity isn't sufficiently rich to express the necessary concepts [Brachman and Levesque, 1985]. Concepts must be manipulated directly if powerful reasoning is to be achieved. The success of any advanced computer-aided tool for enhancing the identification of hazards requires: (i) the development of a representation sufficiently rich to embody advanced scientific concepts, and (ii) a means for manipulating these concepts and reasoning about them directly.

3.1.3 Overview of the Proposed System

We present a new approach to hazard analysis and assessment. An approach that begins with the inductive determination of hazardous chemical or physical reactions and proceeds deductively through the network of processing steps to identify completely all

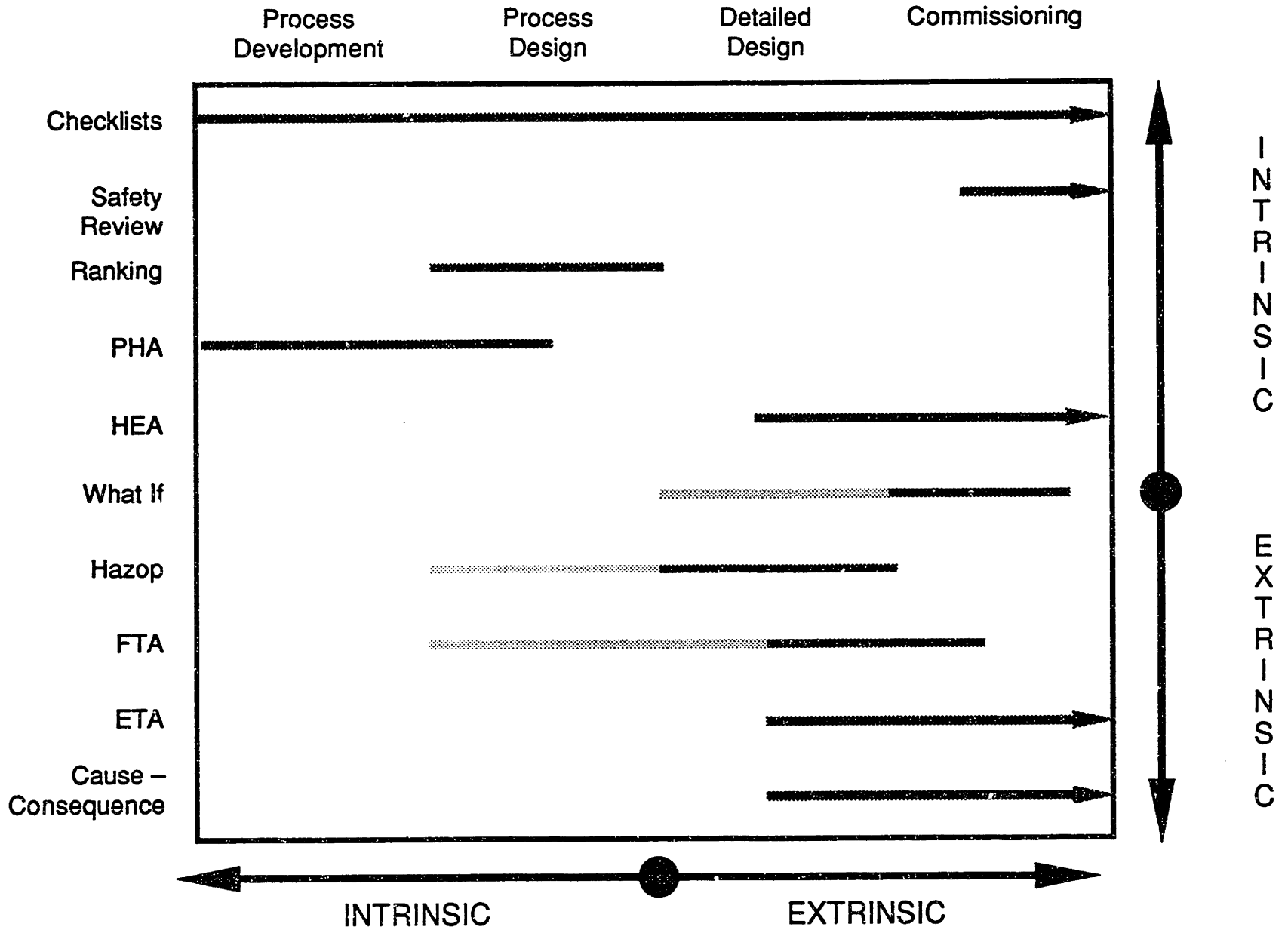


Figure 3-2: Design Phases and Usefulness of Coventional Methods

sources of initiation. The methodology is conceptually distinct from traditional fundamental approaches: it begins with the necessary process interactions that precipitate potential hazards, rather than beginning with process equipment failures and analyzing their effect.

This strategy is more efficient in the identification of reaction based hazards than conventional methodologies. The approach ensures completeness and resolves uncertainty of design quality through first principle quantification; it enables automation. Furthermore, the deductive process beginning at the potential reaction set comes with a guarantee: the set of pathways leading to hazards will be covered by the deductions. Hence, the methodology is complete within the domain of knowledge that identifies feasible reactions.

The methodology is based on two fundamental postulates that formalize and extend Bretherick's observation: "With the exception of releases of toxic or corrosive material, all accidents in storage, handling, or processing of chemicals involve the release of energy at rates too high to be dissipated in the immediate environment without damage." These two postulates establish the theoretical framework that enables the design of a system for automatic identification and analysis of hazards in a decidable manner.

This methodological approach unifies the analysis of potential hazards. As a consequence, the framework can utilize the maximum knowledge available at every point in the design process. It is immaterial whether the design is in the conceptual stage or the final stage. In this spirit, a designer's attention can be focused at the earliest stage to vulnerable areas so that they can be eliminated; and in latter stages, the a priori sequence of events which lead to hazards can be used as an early warning structure [Lees,1983].

In addition, the system proposed in this chapter has been designed to achieve the following major objectives:

(a) Represent a processing system at any level of detail by using multiple coexisting levels of abstraction which can communicate between each other and which can explicitly keep track of interrelated units, and their associated modeling relations.

(b) Automatically generate the set of basic relationships (balances, reaction and transport rates, equilibrium equations, etc) that describe the system. This requirement implies the development of explicitly structured mathematical models, involving variables, terms, and relationship objects.

(c) Capture and utilize qualitative, semi-quantitative relationships (ordinal, order-of-magnitude) or boolean relationships. Such requirement will allow the modeling system to be used beyond the scope of traditional simulators.

(d) Offer explicit documentation of all the hypotheses, assumptions, and simplifications that give rise to a particular model. The model should have all the characteristics of the knowledge it embodies - it should be transmittable from one person to another and open to modification, improvement, and combination with another knowledge. Thus it should allow a direct mapping between ideas of process models and the knowledge of a project (Perkins and Barton, 1987).

The methodology employes the use of domain-specific modeling languages to identify potential reaction based hazards. The utility, development, and usage of these languages is the focus of Section 3.2.

3.1.4 Outline of Chapter

The methodology developed for identifying hazards is divided into two parts. Part I presents the formal framework for inductive hazard identification of hazards, specifies

completeness and efficiency of the approach, presents a modeling language for identifying reaction based hazards, and illustrates the methodology and modeling language interplay (during the identification process) using case studies. Part II presents the deductive component of this methodology: describing how the inductive identification of top level events are used to construct enabling pathways (i.e. root causes), illustrates the methodology through case studies, and describes how the structure of the enabling pathways can altered through appropriate design specifications.

Part I, the focus of this chapter, has been organized as follows: Section 3.2 presents a modeling language for reasoning about chemical systems; Section 3.3 describes the formal methodology and discusses several of its properties; Section 3.4 illustrates the methodology using case studies; Section 3.5 presents a summary of the methodology's characteristics.

3.2. Languages and the Modeling of Chemical Systems

A. N. Whitehead observed that, "By relieving the brain of all unnecessary work, a good notation sets it free to concentrate on more advanced problems". Indeed, by defining appropriate primitives, means of combination, and means of abstraction, engineers have created specialized problem-oriented languages which provide improved modeling facilities for the particular problem at hand [Abelson, Sussman and Sussman, 1985]. For example, the language of "electrical networks" emphasizes the physical modeling of devices in terms of discrete electrical elements (e.g. resistors, capacitors). In contrast, the primitive objects of "electrical systems" are signal processing modules (e.g. filters, amplifiers), underlining the functional behavior of the resulting models. Similarly, in civil engineering we find languages [Maher, 1988, Sriram and Maher, 1986] which allow the declarative modeling of structures (made up of gilders, rods, plates, beams, shells) and their properties (e.g. stress or strain in structural elements). In chemical process engineering we see an evolution in modeling from the language of "unit operations" to

the language of "elementary physical and chemical phenomena". This is due to the inadequacy of unit operations to formalize a large and growing number of specialized processing units.

Domain-specific modeling languages are "very high level" and of special purpose. They have a *theme*, that is, the class of ideas which is optimized to communicate [Hendrikson, 1986]. They allow the user to employ terms and constructs which lie closer to the informal terminology and modes of speech customary in the discussion of domain-specific problems; but, in constructing such a language one should strive for domain-specific generality of the language. Thus, the same modelling language should satisfy all the needs: chemical synthesis, reaction analysis, pathway optimization, and innovative design of reaction or processing networks. The implication of this requirement is clear, *"a domain-specific modeling language should be fully declarative and in no way should its generality be compromised by the specificity of the methodologies of the tasks themselves"*.

3.2.1 System Requirements

Programming efforts outside the domain of hazard analysis have established that expressive, fully declarative specialty modeling languages are indispensable if complex integrated systems capable of synthetic tasks are to be developed. Furthermore, they [Henning, Leone and Stephanopoulos, 1990, Kriticos, 1991] have shown that advanced reasoning and design systems require three essential criteria to be satisfied, namely: (i) all declarative models should be fully articulated, (ii) declarative knowledge should be completely decoupled from procedural knowledge, (iii) a modeling language should be provided to allow the user to think about the task as hand in terms that are familiar.

The system described herein for the automatic identification of hazards and the pathways leading to them satisfies these requirements. The specialized language developed for

reasoning about associated declarative structures is not an ad hoc construction but is based on a clear formalism and satisfies the basic premises of grammar, vocabulary and semantics upon which programming languages are founded.

The remainder of this section will focus on the language we have developed for reasoning about chemical systems. This language is used extensively in the inductive identification of reaction based hazards. The propagation of qualitative and quantitative knowledge through a process flowsheet to deductively identify root causes is presented in chapter 4. Technologies supporting this process have been presented by others [Stephanopoulos, Johnston, Kriticos, Lakshmanan, Mavrovouniotis and Siletti, 1987, Lakshmanan and Stephanopoulos, 1988, Stephanopoulos, Johnston, and Lakshmanan, 1988, Henning, Leone, and Stephanopoulos, 1990].

3.2.2 A Language for Chemical Reasoning

In Section 3.2.1, we highlighted three essential criteria that advanced reasoning and design systems should satisfy. For a computer-aided chemical reasoning system we add a fourth criterion: the logical structure of chemistry should be exploited by imbedding natural constraints into the declarative model.

We propose nine modeling elements for capturing knowledge requisite for chemical reasoning. These elements provide the vehicle for decoupling procedural knowledge from declarative knowledge. Communication between the modeling elements is accomplished through a rich set of semantic relationships that adhere to strict grammatical rules. A class hierarchy is presented for efficiently managing tasks, concepts, and exceptions or rules associated with the modeling elements. The models produced are explicit, transparent and ready to be used without going to literature to check for assumptions, input-output language conventions, and without performing obscure tricks to cope with rigidities. The framework used to formalize our concept of

chemical operations and the language used to manipulate chemical structures, CRL, is presented in Chapter 5.

CRL provides the utility for generating reactions and identifying the resulting pathways, given a set of substrates and a reaction environment. The keyword arguments (i.e. arguments with a colon prefix) accepted by this method are given below:

(FIND-ALL-PATHWAYS :*substrates* :*operators*
:*override-environment* :*initiator-p*)

The user may specify whether an initiator is present or believed to be present. This allows the user to investigate the effect of various reaction trajectories without knowing specifics concerning the initiation mechanism. The keyword :*operations* allows the user to specify the types of transformations to be used to focus the elucidation of pathways.

Alternatively, the user may wish to investigate all theoretically feasible pathways subject to encoded preferences. This is accomplished by supplying :*operators* with $\mathbf{K}^*$, a modified composite operator., and allows the user to investigate pathways having prespecified features (i.e. $\Delta G < 10$ kcal/mol, stereocenters, etc.). Similarly, if there are no preferences, theoretically feasible reactions can be generated by calling FIND-ALL-PATHWAYS with :*operators* $\mathbf{K_{ab\text{-}initio}}$ directly. CRL's functionality and utility is discussed in detail in Chapter 6.

3.3. Proposed Methodology

In Section 3.1.3 we presented an overview of the proposed methodology. In this section we will: establish the framework of the methodology; present a criterion for completeness in identifying hazards; identify properties of the approach; contrast the approach to conventional equipment based methodologies (e.g. Hazop).

In the discussion that follows, we restrict ourselves to hazards whose enabling path involve chemical species. This restriction excludes, for example, hazards (i.e. top level events) created by falls, electrical shock, or impact with stationary objects. We have imposed this restriction because there is not a finite set of hazards or root causes leading to these hazards. Therefore, *to guarantee completeness in the identification of hazards of this type it is an impossible task.* Our imposed restriction does not exclude hazards initiated by these processes, however, provided chemical species are involved. For example, a static charge initiating a chemical reaction or the over heating of a tanks contents, leading to its volatilization and subsequent over pressuring, would be covered.

Our approach is based on two fundamental postulates.

Postulate 1: Hazards can only be created by the interaction of a system through its boundaries or the altering of internal restraints of the system, such that the system proceeds towards a new equilibrium state.

In other words, hazards can only arise through the interaction of two or more states which are not in equilibrium. This postulate follows naturally from the First and Second Laws of Thermodynamics. Since states at equilibrium have no irreversible interactions, a state change is required to proceed towards equilibrium; these changes can only be brought about by interactions through a systems boundary or the altering of internal constraints. Therefore, a state change must accompany a hazard or the system would be at equilibrium. Although many non-equilibrium states may be transgressed in the development of a hazardous system we need only consider the equilibrium states to identify potentially hazardous systems. This is possible because state changes can only be brought about by energy and entropy changes. Since these are state functions, an upper bound can be established on the potential of a hazardous state through the analysis of the equilibrium states leading to it.

Our goal in inductive identification of hazards focuses on the identification of these states. Since any equilibrium state can be characterized completely by *(n+2)* variables, where "*n*" represents the masses of the particular chemical species initially charged and "2" represents two independent variable properties (e.g. temperature and pressure), our task decomposes into the identification of chemical species sets and the independent variable properties that specify the environment of the species. The latter requires the identification of the processing environment, while the former establishes the need to identify the occurrence of various reactions. The identification process is discussed in detail in Sections 3.3.3 and 3.3.4.

While Postulate 1 establishes the framework for inductively identifying hazardous systems, Postulate 2 advances the mechanism by which the pathway of events leading to a hazardous state can be identified deductively:

Postulate 2: The degree of completeness of the set of internal restraints and exogenous factors specifying a hazardous system and its transformation determines the degree of completeness in which the pathways leading to a hazardous state can be identified.

Moreover, Postulate 2 suggests that the deductive identification process is restricted by our understanding of mechanisms which allow states comprising the hazardous system to interact. Since any interaction results in a energy or entropy change of these states, the opportunities for preventing a hazard are limited by the completeness of the restraints and the external variables that specify these states. These in turn are limited by the quality the system description [Battelle, 1985]. But there may not be a finite set of root causes leading to a hazardous state or the identification of a finite set of causes may not be possible (see chapter 4). As a consequence, the pathway of precursor events leading to a hazardous state is incomplete by definition. Despite this property, the constitutive

equations which promote a hazardous system can be used to optimize a design technology's inherent safety. How this is achieved is the focus of chapter 4.

3.3.1 Methodological Framework

Any methodology, whether it is applied in an automatic or manual manner, requires a rich representation of the process if hazards are to be effectively identified. This representation must have sufficient expressive power to allow chains of precursor states/events leading to them to be identified. To satisfy these requirements we have chosen to map the process description into a representational form that allows multi-level multi-faceted process description.

We accomplish this mapping using a specialized modeling language [Kriticos, 1991, Henning, Leone and Stephanopoulos, 1990]. These languages provide a refined description of the process, allowing multi-level modeling of processes with internal consistency (logical and quantitative) among the models that define the process at various abstraction levels. Moreover, they support multiple-context modeling of processes in an automatic or interactive manner. They allow explicit comparison of alternative designs and systematic back-tracking of decisions. Thus, different perspectives of the process such as, structural, topological, and physicochemical relationships can be investigated independently. These languages allow:

a. multiple viewing of process models in terms of structure, topology, and behavior.

b. dissaggregation of abstract models to more detailed ones and aggregation of detailed descriptions to more abstract.

c. contextual description of alternative models for the same process.

d. controlled flow of information among the models at various levels or in various contexts, and detection of modeling conflicts.

e. propagation of qualitative and quantitative knowledge through the defining process models.

Details on the use of these language's is explored by Kritikos and others [Kritikos, 1991, Henning, Leone and Stephanopoulos, 1990].

Utilizing specialized modeling language, such as SYDERELA, we can construct a process description that is suitable for hazards analysis/identification. Our process description is an abstraction of a conventional process representation. It is built on top of a conventional representation giving us access to the functionality of the base representation. This functionality comes with a set of tools that allow us to solve heat and mass balances, identify work interactions, evaluate phase partitioning, etc., and affords the utility to propagate qualitative and quantitative knowledge through the defining network. By focusing our representation on the thermodynamic state description of the process we can facilitate the identification of pathways leading to potential hazards in the most efficient manner. The equipment state space (i.e. the process flowsheet representation) can be mapped to a thermodynamic state space by knowing the trajectory of thermodynamic states comprising a process combined with the transformations that connect these states.

The mapping process that allows us to construct a thermodynamic state based graph representation of a process flowsheet focuses on the state description of the process. The representation is comprised of streams and nodes in accordance with the following definitions:

definition: Streams are idealized flows that connect nodes. They have no accumulation or energy losses.

definition: Nodes are identified as points where discrete thermodynamic transformations occur: a result of changes in intensive or extensive variable values.

These definitions are used to abstract the underlying process representation. As shown in Figure 3-3, SYDERELA allows us to construct an abstract representation utilizing these definitions for our specific application from a conventional process description. This abstract representation has access to any utility, tool, or component that comprises the base representation. For example, the topology of the network is accessed through the graphical construction of the base representation; mathematical relationships such as phase relations, energy balances, and mass balances; mathematical descriptions such as phase equilibria, reaction extent, molar generation rates, mass consumption rates, etc.

Knowledge about the abstract representation and the mapping process are contained in the state description of the process. Figure 3-4 identifies several of the attributes that describe a state. Notice that the state is a composite object: an object comprised of objects. This allows a multi-level, hierarchical representation of a process to be constructed. For example, the operating conditions that specify a state are accessed through the object op-condition-1 which is the attribute value of operating-conditions. Expansion of op-condition-1 identifies the unit in which it is associated as well as the temperature, pressure, flowrate, and composition associated with that unit. This is shown in Figure 3-5.

The chemical species set of a state is the set of chemical constituents that are associated with the system description of that state. The value of the attributes chemical species set, operating conditions, and system volume provide the $(n+2)$ independent variable quantities which are necessary to define a thermodynamic state. An important feature of this description is that each state is described by a vector of *intensive* and *extensive* variables. The intensive vector defines the operational state of the process; while the

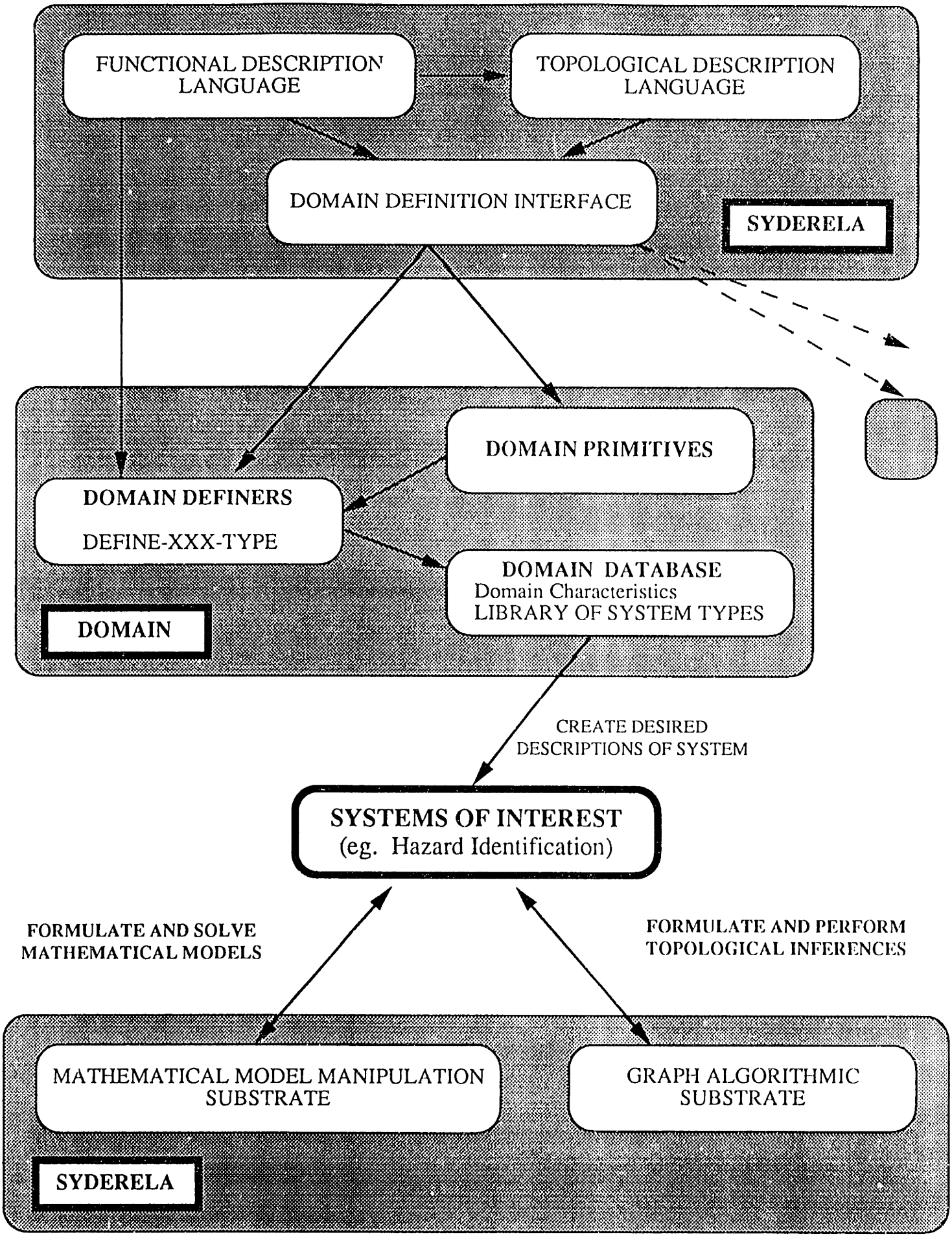


Figure 3-3: Overall Organizaion and Use of SYDERELA

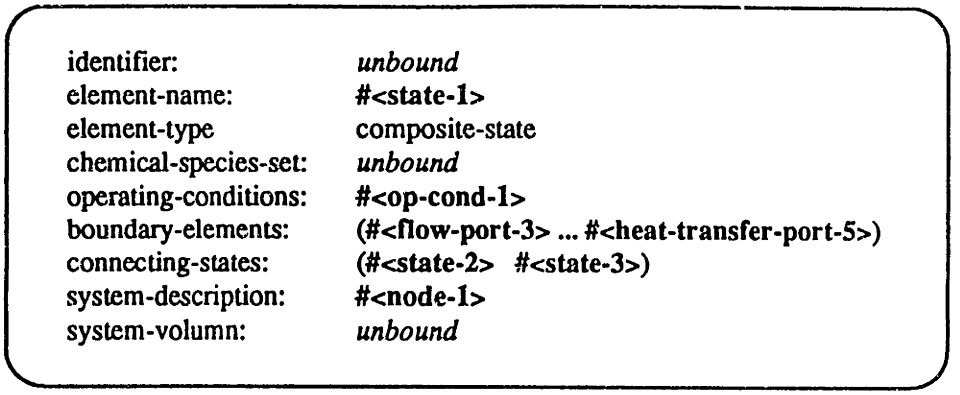


Figure 3-4: Description: state

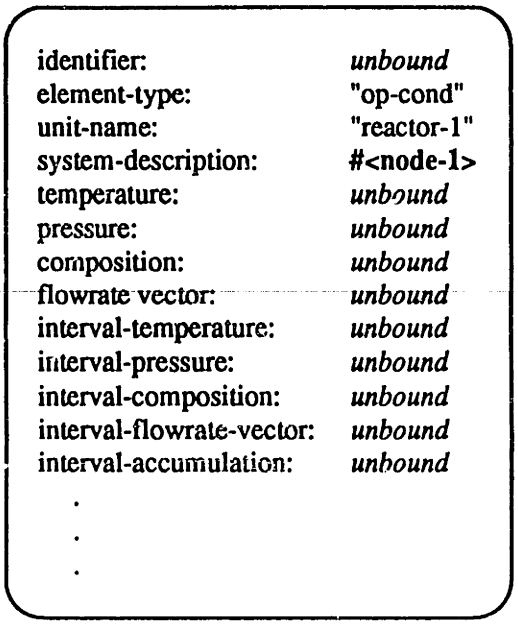


Figure 3-5: Select attributes of op-cond

extensive vector defines the maximum accumulation of a species which can occur. This is bounded by flowrate, reaction rate and physical size of the process equipment. The value of these variables are accessed through the attributes: interval flowrate vector, interval accumulation, and system description. The values bounding an interval are dynamically set. Intervals are defined as:

definition: Intervals are defined are the minimum and maximum allowable value of the variable being described can achieve.

Boundary elements and connecting states are also associated with the state description. Since a state is described as a system and a system has a boundary, boundary elements identify the vehicle for transforming the current state to a new state. The trajectory of connecting states is contained in the attribute value connecting-states. This attribute allows a thermodynamic state representation of a process to be constructed from the individual states which comprise it.

Alternatively, an equilibrium state in which there is no net effect on system boundaries can still be transformed to a new state when the internal restraints which specify the system are altered. This information, as well as the mapping which takes from an equipment based representation to a state based representation, is contained in the attribute value of system-description. An expansion of this state attribute value provides access to the base representation through the **node** description. Figure 3-6 identifies several of the attributes describing the modeling element, **node**. This detailed description allows us to identity system boundary partitions: flow openings, heat transfer openings, work exchange openings, and mass transfer openings. Additionally, the system boundary type is specified as a thermodynamic boundary. The topological system type and topological connector type provide us the links to connect nodes.

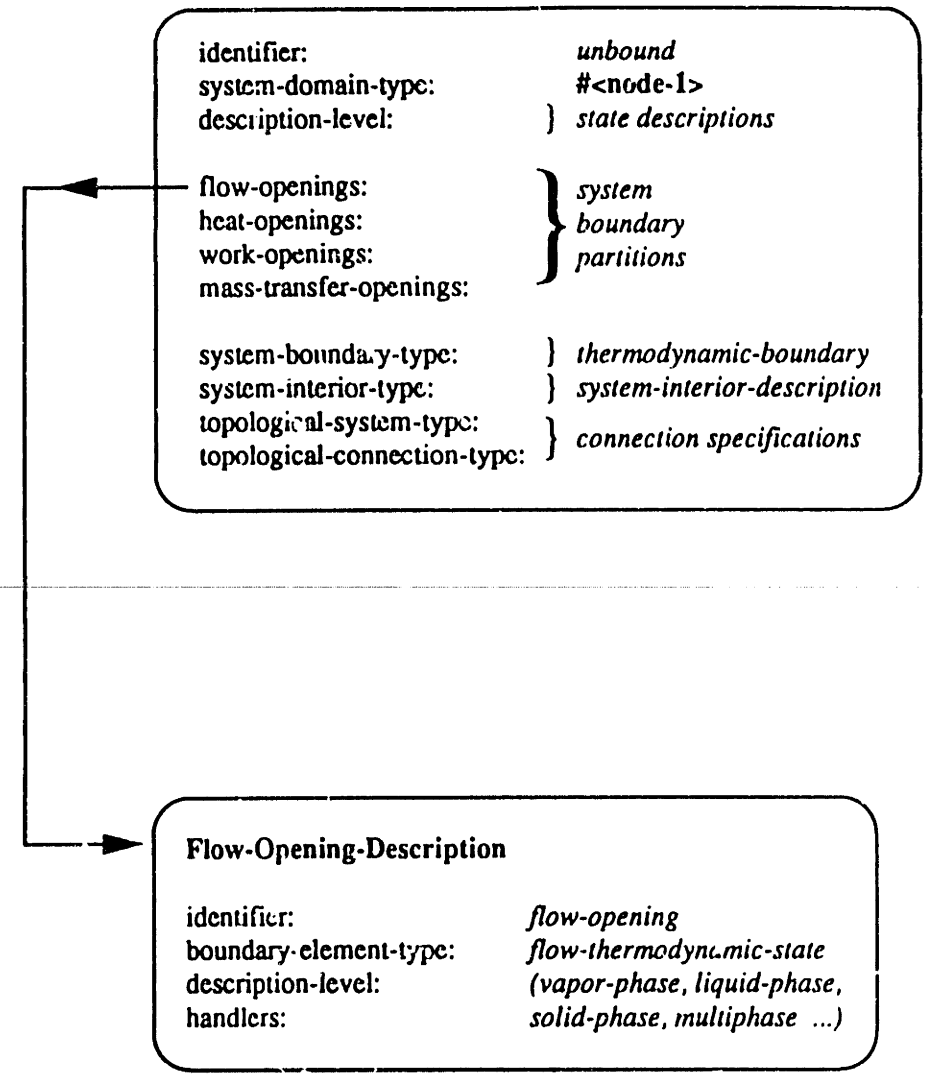


Figure 3-6: Select attributes of node and flow-opening-description

Each of these elements has a description that allows us to evaluate its functionality. In addition, application specific handlers are also provided by the underlying modeling language. For example, the boundary element flow-opening may have a description that requires an understanding of vapor phase, liquid phase, solid-phase, or multi-phase transport. Each of these handlers in turn may access additional methods or handlers to facilitate the evaluation process. Figures 3-7 and 3-8 show the type of descriptions associated with intensive thermodynamic state, a mathematical description of components handler, essential reacting state, and a mathematical description of reactions handler, respectively. These figures give an indication of the type of variables, variable arrays, relations and relation arrays that are required for elucidating such a description.

The attribute value system-interior-type specifies the interior system description of the **node**. The value of this attribute allows us to map the node representation to the process equipment representation. Notice that it is the attributes of the **node** in combination with the features of the underlying specialized modeling language that afford a multi-level, multi-faceted view of the process. Hence, we are able to move independently from the node representation to the equipment representation and pursue, as needed, analysis of momentum, mass, and energy interactions of the defining network. Later we will show that we also have access to the representation specifying the chemical constituents which are associated with the state. It is the utility of the specialized modeling languages from which these base representations were constructed that allow access to their various elements: CRL in the case of chemical species; SYDERELA for the node representation. In the sections that follow, we will show how the information contained at the equipment level is accessed by the chemical species to assist in the evaluation of potential reaction alternatives.

An example of a mapping from an equipment state representation to a thermodynamic state representation is shown in Figure 3-9 for an isothermal vertical packed bed catalytic

Mathematical-description: intensive-thermodynamic-state

variables:	temperature
	pressure
	molar-enthalpy
	mass-enthalpy
	molar-entropy
	mass-entropy
	moalr-internal-energy
	mass-internal-energy
	molar-gibbs-free-energy
	mass-gibbs-free-energy
	molar-helmholtz-free-energy
	mass-helmholtz-free-energy
	molar-heat-capacity
	mass-heat-capacity
	molar-volume
	mass-volume
	density
	molar-density
variable-arrays:	mole-fraction-vector
	mass-fraction-vector
	partial-molar-enthalpy-vector
	partial-mass-enthalpy-vector
	partial-molar-entropy-vector
	partial-mass-entropy-vector
	partial-molar-gibbs-free-energy-vecotr
	partial-molar-helmholtz-free-energy-vector
	partial-molar-heat-capacity-vector
	fugacity-vector
	partial-mass-heat-capacity-vector
.	.
.	.
.	.

Figure 3-7: Description: intensive-thermodynamic-state

Mathematical-description essential-reacting-state

variables:	molar-generation-flowrate mass-generation-flowrate molar-consumption-flowrate mass-consumption-flowrate
variable-arrays:	molar-reaction-extent-vector mass-reaction-extent-vector molar-generation-vector mass-generation-vector molar-consumption-vector mass-consumption-vector
relations:	overall-molar-generation-definition overall-mass-generation-definition overall-molar-consumption-definition overall-mass-consumption-definition
relation-arrays:	molar-generation-definition-vector mass-generation-definition-vector molar-consumption-definition-vector mass-consumption-definition-vector
.	.
.	.
.	.

Figure 3-8: Description: essential-reacting-state

reactor equipped with temperature and pressure sensors, and explosion vent and a distributor plate. Notice that the equipment and sensors are not associated with the state representation. They are contained in the base representation; but reside in the equipment level. As discussed earlier, flow, work, heat, and mass interactions are all modeled independently. This allows us to evaluate independently the effect of these processes. Independent evaluation assists in the identification, evaluation, and assessment pathways leading to hazardous states.

3.3.1.1 State Generation

Using this representation, we now focus on the identification of associated thermodynamic states. Since new thermodynamic states are created by reactions, a means for generating potential reactions between species resident in the node description (i.e. the chemical species set) is required. We assess the likelihood of reaction as well as the inherent instability of each species associated with a node using CRL. The generation of infeasible species is limited by the knowledge embedded in **reaction-environment** and **K**, $\mathbf{K}^*$, or $\mathbf{K}_{\textbf{ab initio}}$.

Each reaction is assessed in the context of the processing environment (i. e. the reaction environment) that establishes limits of operation for the defining node. FIND-ALL-PATHWAYS accesses the value of the **abc** attribute environment and hence the values specifying this condition using the semantic relationships of CRL (see Chapter 4).

For example, FIND-ALL-PATHWAYS applied to a chemical species set, CSS, (i.e the set of chemical specified bound to a **state**) containing three species evaluates all combinations. That is, the set **{A B C}** requires evaluation of the following sequences:

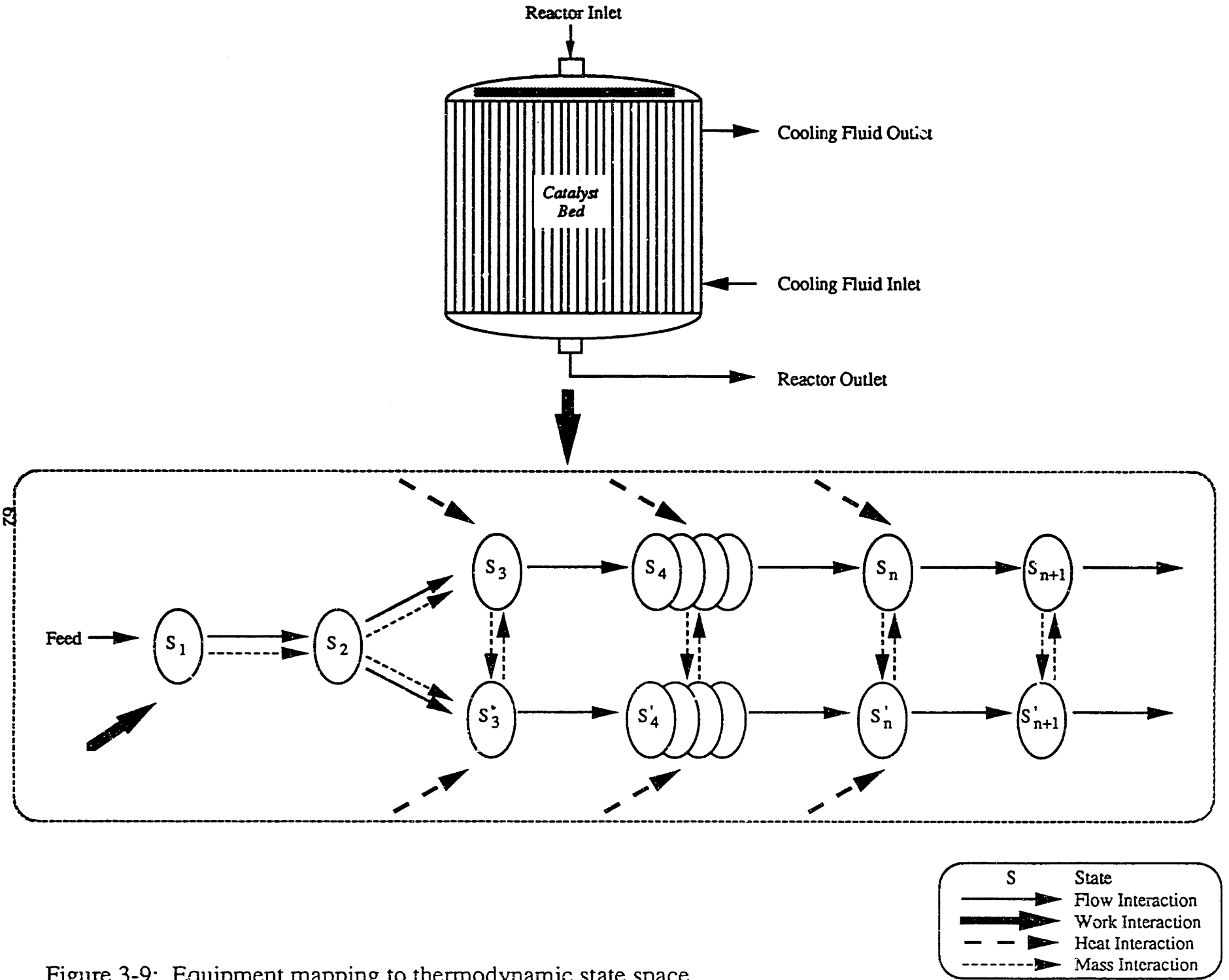


Figure 3-9: Equipment mapping to thermodynamic state space

$$
\begin{aligned}
\mathbf{A} &\rightarrow ? \\
\mathbf{B} &\rightarrow ? \\
\mathbf{C} &\rightarrow ? \\
\mathbf{A+B} &\rightarrow ? \\
\mathbf{A+C} &\rightarrow ? \\
\mathbf{B+C} &\rightarrow ? \\
\mathbf{A+B+C} &\rightarrow ?
\end{aligned}
$$

Unique products generated by these reactions are added to the CSS and evaluated. A single call to FIND-ALL-PATHWAYS performs the global evaluation:

(FIND-ALL-PATHWAYS *:substrates* (A B C) *:operators* $\mathbf{K^*}$)

As in SYDERELA, CRL provides objects to contain the information generated by these various evaluations. For example, the attributes describing a reaction are shown in Figure 3-10. These attributes describe not only the reactives in products of the reaction, but also establish the reaction environment enabling conditions composing transformations. Furthermore, the utilities of CRL provide additional description of reaction and may include reaction stoichiometry, equilibrium constant, rate expression, and identification of competing reactions. Similarly, Figure 3-11 shows the attributes describing the object pathway.

The **abc** representation advanced by CRL provides direct access to the **state** representation through the **abc** attribute environment. The value of the attribute environment (see Figure 3-12) provides the link to the surrounding environment in which an **abc** is exposed. An expansion of this attribute value is shown in Figure 3-13. Notice that the value of attribute system-element provides the link between the **state** representation, which was shown in Figure 3-4, and the **abc** representation.

Reaction Object

identifier:	*unbound*
name:	initiation-1
reactants:	(# <Cl_2>)
products:	(#<Cl> #<Cl>)
stoichiometry:	((#<Cl_2> . -1) (#<Cl> . 2))
reaction-environment:	#<reaction-environment-1>
enabling-conditions:	K_f
composing-transformations:	K_t
composing-reactions:	*unbound*
rate-expression:	#<rate-expression-1>
equilibrium-constant:	#<equilibrium-constant-1>
context:	*unbound*

Figure 3-10: Description: reaction object

Pathway Object

identifier:	*unbound*
name:	pathway-1
reactants:	(# <C_2H_6O> #<Cl_2>)
products:	(#<$C_2H_5C_{10}$>)
stoichiometry:	*unbound*
competing-pathways:	(<pathway-2> #<pathway-3>)
composing-reactions:	(#<initiation-1> #<abstraction-1> #<combination-1> ...)
global-rate-expression:	#<composite-rate-exp-1>
global-equilibrium-constant:	*unbound*

Figure 3-11: Description: pathway object

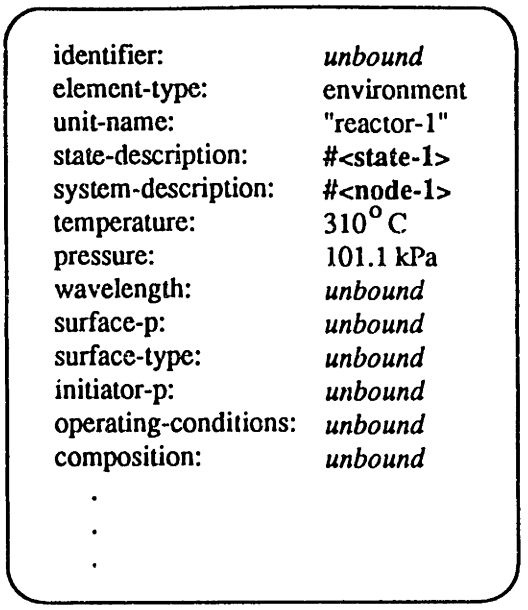


Figure 3-12: Description: reaction-environment

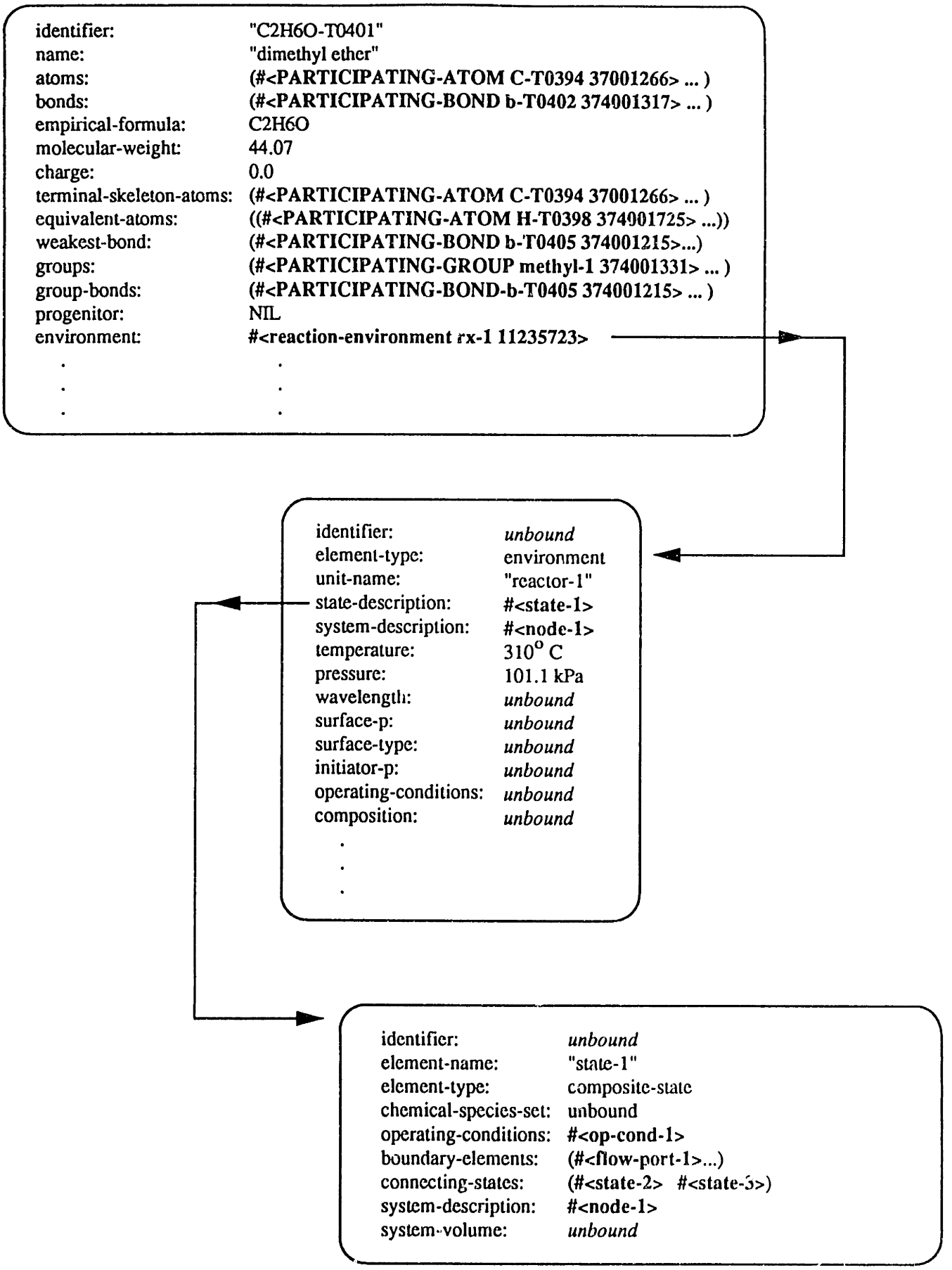


Figure 3-13: Accessing process representation from chemical representation

Figure 3-13 shows one access route from the **abc** representation to the **node** representation and on to the underlying process representation. This link allows any chemical species associated with the process, to be accessed using the semantic relationships of CRL. Similarly, the semantic relationships of the process modeling language afford access to the information contained in the underlying base representation. *Together, these specialized modeling languages allow functional and topological information derived at one point in the process to be accessible from any other point in the process.*

3.3.1.2 Hazard Identification

Using these tools and the representations that are built with them, potential hazards are inductively identified by combining various chemical and physical environments of the process in an attempt to generate new process states. The types and numbers of process states are determined by both the process configuration and the operating conditions. Enabling conditions of a potential hazard are identified by establishing the conditions specifying the thermodynamic states which precede it. These conditions can be either intended or induced, resulting from changing boundary elements, system descriptions, or reacting states. Normally, they manifest themselves as procedural, material, or boundary changes.

A potential hazard is said to exist when an environment or a sequence of environments cannot be dissipated or prevented by the process. Using this environment as a goal state the sequence of process and/or operational changes which led to its creation can then be identified deductively. Consequently, root causes and their temporal preconditions (whether they be equipment and/or operational failures) can be identified and associated with the known hazard. Evaluation of the pathways leading to the hazard permit quantification of the top level event and affords an assessment of the potential hazard in the context of the design technology. In chapter 4, we will show how the structure of the

pathway itself provides a metric for the inherent safety for the design technology being evaluated.

3.3.1.2.3 information flow: inductive identification

Information flow during the execution of this methodology is shown in Figure 3-14. Two components comprise the methodology: inductive identification of reactive states; deductive identification of pathways leading to those states. The inductive phase begins with a process description from which reactions are identified using CRL and the tools described in section 3.2.2.6. These tools allow chemical reacting states to be identified. Once the entire chemical species set is elucidated, a **node** can be fully described and the **states** disseminating from it established (i.e. the $n+2$ independent variable quantities are known). With this description potential physical reactions (e.g. overpressurizing, overheating, overcooling, etc.) can be elucidated using classical thermodynamic (open and closed system treatment) and transport analysis of **node** descriptions. The utilities of the underlying specialized modeling language afford evaluation of these phenomena.

Potential physical reactions can only be identified with any assurance of completeness *after* the complete specification of the chemical species set, CSS. This is a result of Postulate 1: $(n+2)$ defining variables must be known before a thermodynamic state can be specified. Since incomplete specification of "n" leads to incomplete specification of potential states, a complete node description (i.e. the total CSS must be identified) is required before physical reactions can be identified completely.

Once potential reactions are identified, they are posted on a potential reaction list. The promoters of each reaction are then identified by tracing the variable values that led to their creation. This is achieved using the links provided in the thermodynamic state representation and the **abc** representation. Similarly, by accessing the information contained in **K**, $\mathbf{K}^*$, or $\mathbf{K}_{\mathbf{ab\ initio}}$, the conditions which promoted a particular chemical

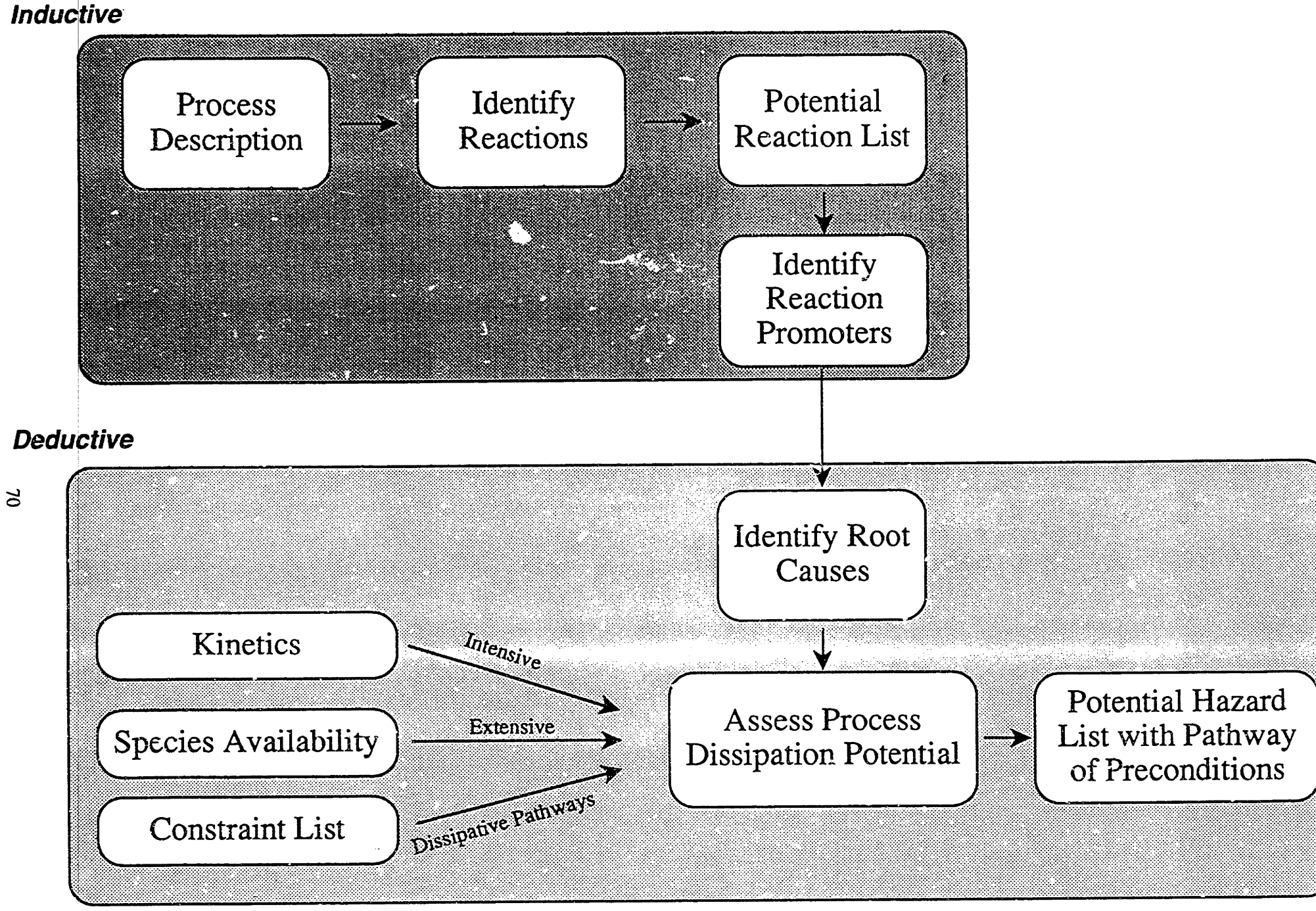


Figure 3-14: Implementation of Methodology

reaction can also be identified. These conditions may be chemically induced, as in the case of nucleophilicity, or physically induced as in the case of high energy environments enabling radical or photochemistry, or both. These promoters are then used to establish the underlying pathways that enable the top level event.

3.3.1.2.4 information flow: deductive identification

The deductive identification of pathways leading to a top level event is accomplished by tracing the effect of the variables which describe the state preceding the event. The influence of these variables is identified using the network of equations that specify the process. The variable influence trace completely identifies the pathways leading to the top level event within the scope of the modeling effort. The structure of these equations: how the variable values are satisfied; establishes a design technology's inherent safety.

By propagating the values of the enabling/promoting conditions through the network, the process's ability to dissipate potentially hazardous effects can be assessed using the knowledge and utilities contained in the supporting process and chemical representations. For example, the variable values may be captured by CRL, as in the case of chemical kinetics, or by the underlying process representation, as in the case of temperature, pressure, and flowrate. Although the knowledge required to assess dissipation potential is often held in different abstractions of the overall representation, the semantic relations of modeling languages afford efficient access to this information independent of its position.

The effect of protective processes, equipment restraints, sensors, emergency procedures, etc. are captured as constraints. Constraints may be embedded in the underlying representation as equations; or associated directly to it via a constraint list:

definition: A constraint list is a collection of explicit process restraints. These restraints may be *passive*, such as materials of

construction, or *active* such as, protection processes, sensors, and operating procedures.

Constraints limit the variable value of an enabling condition. They may achieve this task in one of two ways: demand mitigation or demand prevention .

Mitigation requires no corrective action to limit a variable value. This is accomplished by designing equipment to restrain the top level event, given an enabling condition, or by limiting the value that an extensive or intensive variable can achieve. For example, the value of an intensive variable can be limited by the physical phenomena that is allowed to occur within a process. Orchestration of this phenomena: where and how it occurs; can place a restriction on the species which may be generated, the rates at which reactions occur, the separation of constituents into phases, etc. Similarly, extensive variable values such as equipment volume, maximum accumulation, or total mass, can be limited by process equipment design. Since the equations describing a process are a manifestation of this phenomena, select modification of the pathway can maximize the inherent safety of the design technology. More importantly, *modification of the pathway topology or the achievable variable values is the only means by which the inherent safety of a design technology can be altered.*

Prevention of a demand requires a corrective action be taken. These actions are designed to limit the variable value of an enabling condition; but, it does so through active participation of protective equipment (eg. release vents), control loops, or operating procedures. Like demand mitigation, demand prevention necessitates an understanding of the cause and effect link and hence, the underlying physical phenomena. However, we have chosen to treat explicitly the equations modeling this phenomena to facilitate detection of events enabled by protective mechanism failure.

This combined approach: the inductive identification of top level events and the deductive identification of enabling pathways; allows potential hazards and the sequence of events leading to them to be identified completely within the scope of the modeling effort. Moreover, since the top level event is generated from its underlying states, the pathway of preconditions and the temporal ordering of those preconditions are explicit spanning only the minimum cut set.

3.3.1.3 Hazard Identification Algorithm

The algorithms used in this approach are expressed below, in pseudocode convention [Cormen, Leiserson and Rivest, 1990]. Therein, a terminal node is defined as:

> **definition:** A terminal-node is the first node a feed enters or the last node a product or byproduct exits in the process flowsheet.

The routine calls for two loops. This is necessary to simulate multiple simultaneously occurring boundary failures.

```
<GLOBAL-HAZARD-IDENTIFICATION>
    input: process-flowsheet
    initialize
        process-node-representation ← apply MAKE-PROCESS-TRANSFORMATION
                                      to process-flowsheet
        terminal-nodes ← apply FIND-ALL-TERMINAL-NODES to process-node-
                         representation
        potential-hazard-list ← nil
    for each node in process-node-representation
        potential-hazard-list ← append (apply IDENTIFY-POTENTIAL-HAZARD to
                                (node process-flowsheet)) to potential-hazard-list
        return
    for each node in terminal-nodes
        expand node-scope
          until UNIQUE-STATE-IDENTIFIED-P
            or EXPANSION-NOT-POSSIBLE
          then
```

```
        potential-hazard-list ← append (apply IDENTIFY-POTENTIAL-HAZARD
                                to (node process-flowsheet)) to potential-
                                hazard-list
    return
end
return
```

The internal routine, IDENTIFY-POTENTIAL-HAZARD, called from GLOBAL-HAZARD-IDENTIFICATION, returns a potential hazard and its list of enabling conditions. The algorithm for IDENTIFY-POTENTIAL-HAZARD, expressed in pseudocode, is given below:

```
<IDENTIFY-POTENTIAL-HAZARD>
    input: node
           process-flowsheet
    initialize
        potential-reaction-list ← nil
        associated-states ← nil
        extended-CSS ← nil
        potential-reaction-list ← apply FIND-ALL-PATHWAYS to CSS of node
        extended-chemical-species-set ← collect UNIQUE-CHEMICAL-
                                        CONSTITUENTS from
                                        potential-reaction-list
        associated-states ← apply EVALUATE-STATES to node and
                            extended-CSS
    for each state in associated-states
        when UNIQUE-STATE-DESCRIPTION-P
            potential-reaction-list ← apply EVALUATE-PHYSICAL-
                                      REACTIONS to state
            return
        return
    for each reaction in potential-reaction-list
        enabling-criteria ← collect FIND-ENABLING-CRITERIA
        classified-influence-paths ← apply CONSTRUCT-VARIABLE-INFLUENCE-
                                     PATHWAYS to (enabling-criteria process-
                                     flowsheet)
        root-causes ← apply IDENTIFY-NONDISSIPATIVE-PATHWAYS to classified-
                      influence-paths
        when root-causes
            potential-hazard ← list reaction enabling-criteria root-causes
            return
        return
```

end
return

Notice the interplay between the methodology and the specialized modeling languages and the importance of that interplay. These languages enable the methodological approach. For example, FIND-ALL-PATHWAYS applied to a chemical species set utilizes the semantic relationships of CRL to construct potential-reaction-list. Similarly, the representation utilized by CRL allows enabling-criteria to be explicitly associated with each reaction; these criteria can come from the **state** representation or from the inherent physicochemical properties associated with the chemical constituent itself. Likewise, EVALUATE-PHYSICAL-REACTION utilizes functionality of the base representation to assess possible effects and to identify enabling criteria. These criteria come from the operating environment of the process equipment.

By interweaving the semantic relationships of the two modeling languages throughout the methodology, we are able to interplay the respective representations to satisfy the representational needs of the task at hand. For example, root causes are identified by propagating (i.e. solving) the enabling criteria through the network of equations lying in the base representation and defining the process flowsheet. These criteria, however, are located in different representations, at different abstraction levels, to afford the resolution necessary to identify potentially hazardous conditions. The specialized modeling languages provide the tools to map between different representational forms and identify potential hazards with their enabling conditions and underlying root causes in an efficient manner.

Two additional features of this algorithm are: (a) interval values are *dynamically* redefined whenever a condition is found to exceed the existing value (values may be exceeded in the positive or negative direction), and, (b) a node's description (i.e. the

scope of the defining system) can be expanded to simulate common boundary failure or process equipment malfunction. Both features are discussed in detail in chapter 4.

3.3.2 Completeness in Hazard Identification

We now establish the *criterion for completeness* for any hazard identification methodology. [Herein, completeness refers to the ability of a methodology to identify all possible top level events (i.e. hazards).] The methodology presented in Section 3.3.3, establishes how we use this criterion to completely and systematically identify hazards in an *efficient* manner.

Theorem: Any hazard identification methodology must cover all subsets of sources present in a process network to guarantee completeness.

Proof: In Appendix A, we present a constructive proof to establish that any arbitrary set of sources can be assembled at some point in a process network. The algorithm advanced establishes this mechanism. Since it is *a priori* impossible to know that a particular subset of materials is non-hazardous under all conditions, any method having the property of completeness must be able to generate and evaluate each of the subsets.

Corollary: Equipment based methodologies can be complete.

Proof: The constructive methods presented in Appendix A, involve specifying presence or absence of flow and direction of flow, through connections between pieces of equipment and within equipment. Thus, if an equipment based methodology is sufficiently detailed to consider the temporal order of failures, and the possibility of equipment items experiencing arbitrary patterns of states, it will be complete.

3.3.3 Properties of Reaction Based Hazard Identification

We now consider the relative efficiency of different methods. Since it is meaningless to compare the efficiency of methods that are incomplete versus those that are complete, for the purposes of this section we consider only complete methodologies. But before beginning we point out one further result that can be derived from the completeness analysis presented in Section 3.3.2:

Theorem: There is no hazard's analysis method that is both complete and whose running time is bounded by a polynomial in the number of initial species present in the plant.

Proof: The possible number of subsets of species is $2^{|S_0|}$ where S_0 is the set of initial species in the plant. Any complete method must at least enumerate these sets, therefore its running time is lower bounded by $2^{|S_0|}$ which is not bounded by any polynomial in S_0.

For the purposes of this chapter, an equipment based method is characterized by two parameters, **G**, **D**, and classified according to the way the method handles temporal order:

Parameters:

G ≡ Set of Guidewords applied (e.g. more, less, part of, normal, no)

D ≡ Set of Features of the Equipment, (e.g. flow, temperature, pressure,concentration, etc.)

Method Classification:

Atemporal: In atemporal methods, the equipment is failed according to the above guidewords, but no attempt to consider the effect of temporal order of the failures is made.

Temporal in Single Failure Mode: In these methods, the temporal order of failures is considered, but each piece of equipment is allowed to leave its normal state and enter only a single failure state.

Temporal versus Bounded Failure Mode: This is the same as the above, but each equipment item is allowed to enter any of its defined states a maximum of κ times.

Now we define an Equipment State as a set of guidewords, one for each feature. The Equipment State Space is the set of all possible *states* of the plant and is fully defined by the state of all pieces of equipment in the plant. Calculation of the various sizes of equipment space engendered by the three classes of equipment based methods is now possible.

First let us define:

$\Omega_e \equiv$ The set of possible equipment states of equipment

then:

$$|\Omega_e| = \prod_i G_{ei}^{D_{ei}}$$

where:

$G_{ei} \equiv$ The set of guidewords for feature i of equipment item e

D_{ei} ≡ The set of features of equipment item *e*

To simplify the presentation, we will assume that all pieces of equipment have the same set of features and guidewords, and that the modulus signs on the sets is implied. Using this assumption, equipment state space size is classified below by temporal order:

For an atemporal equipment state space, where E represents the equipment in the plant, we have:

E ≡ The set of pieces of equipment in the plant

then,

Size of Atemporal Equipment State Space: $\Omega^E = \delta_p$

It is assumed that we can sequentially order the failures; we define a time slot by one piece of equipment changing its state. For the temporal equipment space, we have E time slots, since the plant can change state at most E times. In the first time slot, we have E pieces of equipment that can enter (Ω-1) states (i.e. (Ω-1) because there must be a change in state). In the second time slot, there are E-1 pieces of equipment, in third E-2, etc.

Size of Temporal Equipment State Space: $= [(\Omega-1)E][(\Omega-1)(E-1)]$

$= (\Omega-1)^E E!$

In bounded failure mode we allow the simultaneous failure of pieces of equipment, but bound the number of times they may change state by a fixed parameter κ. Allowing the repetition of states this leads to:

Size of Bounded Failure Mode: $(\delta_p)^\kappa$

An equipment based method then systematically enumerates its space of equipment failures, deducing the set of chemical species present in the equipment based on the equipment state.

3.3.3.1 Efficiency Comparison

First it is necessary to establish the meaning of one method being more of less efficient than another. There are at least two possible dimensions along which comparisons can be made, time and space.

definition: $>_t$ is a binary relation over hazard analysis methods, $M_1 >_t M_2$ if and only if M_1, requires less computational time on *some* hazards problem class but M_2 *never* requires less computational time on *any* problem class.

definition: $>_s$ is a binary relation over hazard analysis methods, $M_1 >_t M_2$ if and only if M_1, requires less resource of computer memory (i.e. space) on *some* hazards problem class but M_2 *never* requires less resource of computer memory on *any* problem class.

Theorem: C1 $>_t$ E: A reaction based method is more efficient than any equipment based method.

Proof: From the characterization of the two methodologies, we can draw the following two pictures (see Figure 3-15).

From the point at which the species set is generated, the equipment based methodology is identical to a reaction based methodology. The implication of this statement is that the only information generated by the equipment analysis which is used in the identification of top level events is the state space; which collapses to the species set since temperature and pressure are local variables.

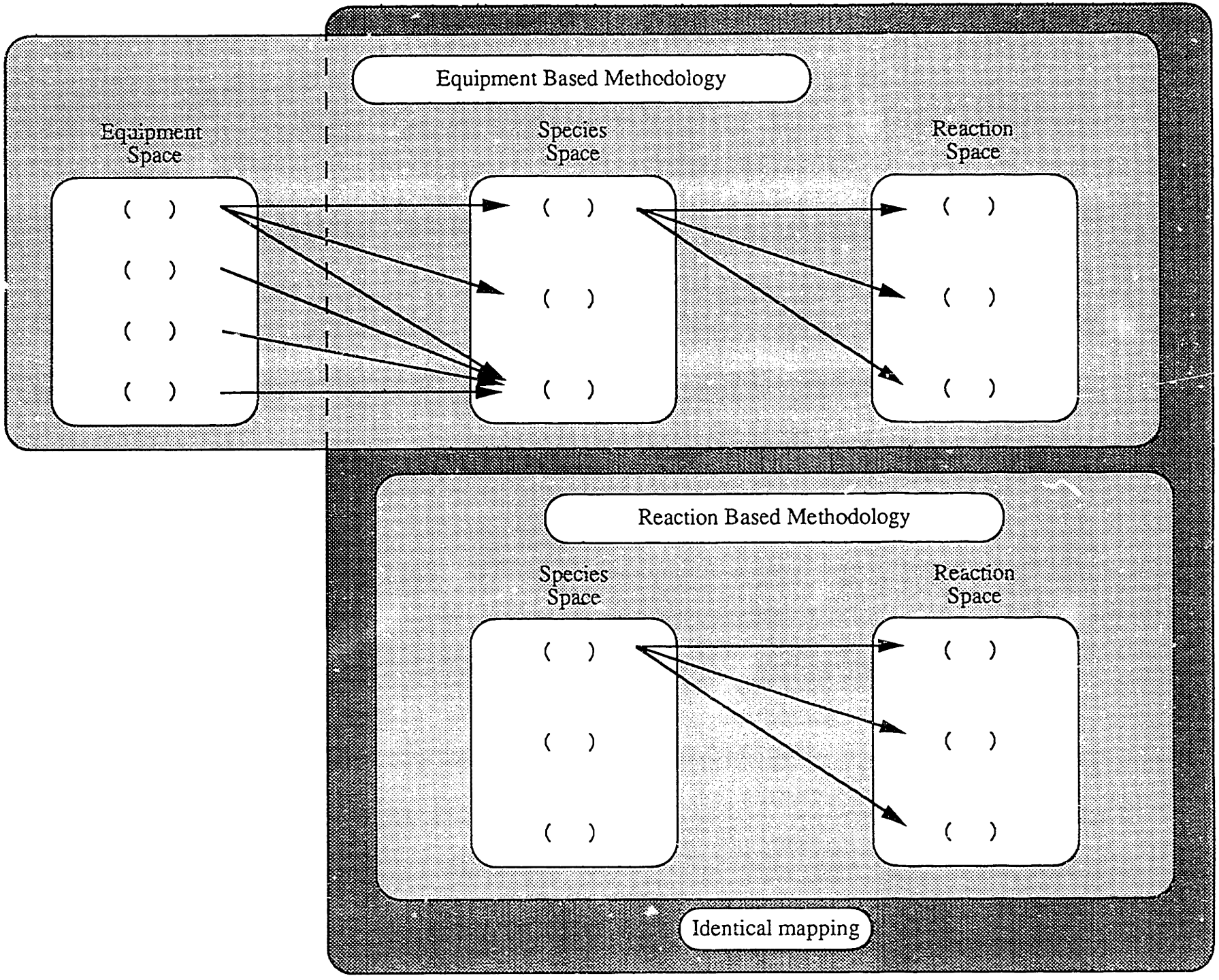


Figure 3-15: Mapping of equipment space to species space

Statement: It is possible for the mapping from the equipment space to the species space to be one to many. Clearly different species sets can be generated at different points in the plant, or at different times in the same place. The maximum cardinality of the mapping is: E in the case of atemporal; E^2 in temporal; κE in finite temporal.

Statement: It is possible for the mapping from the species space to the equipment space to be one to many. Again, the same species set could be generated in several different plant states, or series of plant states.

The only difference between the two methods is in the generation of the species set. The equipment based method generates the set *indirectly* via the equipment state space; a reaction based method generates it *directly*. Since both approaches must generate every element for completeness of the set, the difference in efficiency between the equipment based method and the reaction based method must result from generating an element more than one. This must always be the case if the size of the equipment space (ES) is larger than that of the species set space (SS):

$$ES > SS$$

In an atemporal methodology:

$$\Omega^E > 2^{|S_0|}$$

We can bound the minimum size of ES, note $G^D \geq 2$, that is every piece of equipment is capable of failing:

$$ES \geq 2^E$$

Further, we can examine some special cases where $ES \geq SS$ will hold. If there exists a separate source for each species, the $|E| \geq |S|$.

We now focus on two key aspects which differentiate the equipment based approach from the reaction based approach: mapping efficiency and resolution. As established earlier, any complete methodology must identify every potential thermodynamic state. The complexity encountered in this enumeration is 2^{S_o}.

Revisiting the complexity of the equipment based approach we have:

$$\Omega^E = G^{D^E}$$

but we can lower bound **G** by 2, indicating presence or absence, and **D** by n, the masses present in the system. This substitution yields:

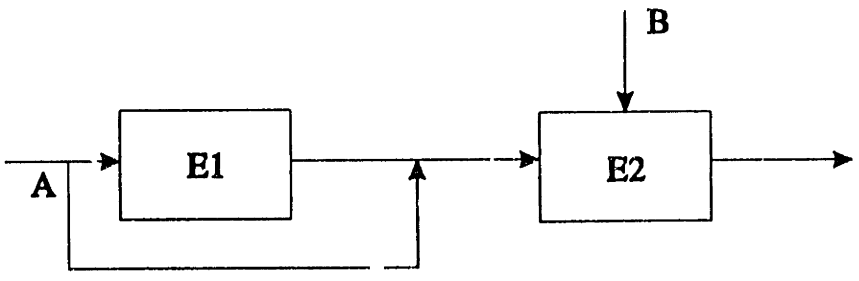
$$\Omega^E = 2^{n^E}$$

however, since a lower bound on n is S_0 we have:

$$\Omega^E = 2^{S_o{}^E}$$

In other words, the equipment based approach has embedded the criterion for completeness and for a single piece of equipment matches the complexity of the reaction based approach. However, because of the mechanism that the equipment state approach uses for generating these states, when multiple pieces of equipment are involved redundant states can be generated making the approach less efficient.

For example, consider the following illustration involving the leaking of A in E1, followed by mixing A and B in E2:



where:

$\mathbf{G} \equiv \{\text{Flow, No Flow}\}$

$\mathbf{D} \equiv \{\text{Species A, Species B}\}$

$\mathbf{E} \equiv \{E1, E2\}$

Thus the equipment space size is:

Equipment Space: $2^{2^2} = 16$

but, the species space is:

Species Space: $2^2 = 4$

If we define:

$\bar{a}_i \equiv$ No flow of "a" to E_i

then we can establish the mapping from the equipment space to the species space. A subset of the mapping from the equipment space to the species space is shown in Figure 3-16.

This illustrates the many-to-many nature of the mapping process. The mapping process is a source of inefficiency in equipment based approaches.

But, the equipment based approach is also engineered to perform a second task: identification of root causes. Any complexity in addition to 2^{S_O} is an effect of this objective:

$$(2 + c_1)(S_O + c_2)^E$$

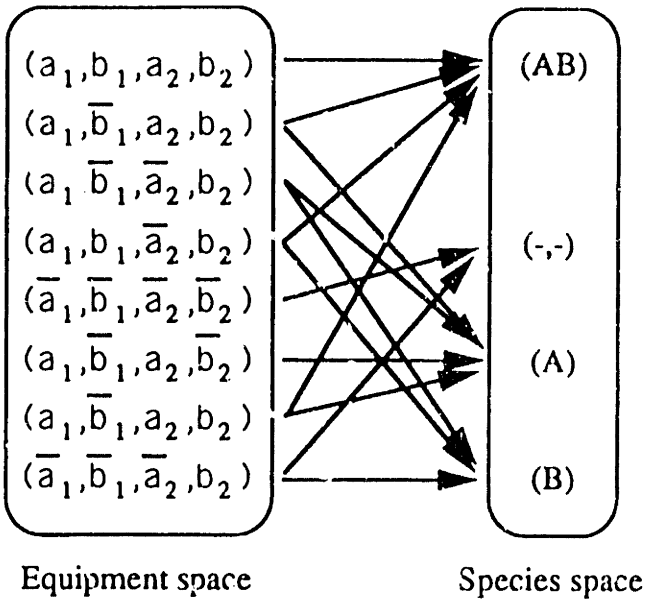


Figure 3-16: Many to one mapping of equipment space to species space

where c_1 represents additional guidewords and c_2 represents additional process parameters. Under normal conditions **G** may take a value of 5; **D** a value of 4; and E a value representing the major pieces of equipment. If we allow E to be of order 10 then we have:

Equipment Space: $5^{5^{10}} \approx 9 \times 10^{34}$

This is the number of systems that must be identified to identify the potential hazards associated with the process. Notice that this does not guarantee completeness. To achieve completeness, **D** must span S_0. Assuming five initial species:

$$\mathbf{D} = (S_0 + c_1) = (5 + 4) = 9$$

and,

Equipment Space: $5^{9^{10}} \approx 8 \times 10^{62}$

In other words, 8×10^{62} different systems must be evaluated to establish completeness in the identification of top level events (i.e. potential hazards). However, as shown in Section 3.3.0, completeness cannot be guaranteed in the identification of root causes. In fact, due to the approach employed by equipment based hazard identification, there is no guarantee that all states will be generated. This is because **G** and **D** are designed to trace out *causes* not generate states. This guarantee only comes when the resolution of the tracing mechanism (i.e. the resolution of **G** and **D**) completely defines the enabling state of the potential hazard.

Moreover, although G^D may have substantially greater complexity than 2^{S_0} this does not ensure that all states will be identified. For example, to identify a potential hazard that is caused by changing the catalyst shape the equipment based approach must include surface area as a process parameter in **D**; a reaction based approach has this information

embedded via the rate expression. Thus, as a consequence of its formulation and the undecidedly of conditional operators (see chapter 4), the equipment based approach may not guarantee complete identification of potential hazards and can not guarantee the complete identification of root causes leading to those hazards.

3.4. Examples in Reaction Based Hazard Identification

In this section, we will demonstrate how the methodology can be used to identify hazards through a series of case studies. We will focus on the detection of potential hazards arising from changing operating conditions rather than equipment malfunction or failure; the latter is discussed in Chapter 4.

3.4.1 Case Study 1: sodium hypochlorite

Consider, for example, the evaluation of the discharge system shown in Figure 3-17. GLOBAL-HAZARD-IDENTIFICATION begins by transforming the simple process diagram which consists of a single mixing junction, into a single process node representation. Therein, the discharges which feed into the mixing junction are labeled stream-1 and stream-2, respectively. The **node** describing this system in Figure 3-18. The description level describing this **node** focuses on the flow state of the system Flow openings are specified for **stream-1**, **stream-2**, and **stream-3**. The system interior is defined by the union of **stream-1** and **stream-2**; and the conditions associated with those streams. This is shown in Figure 3-19. Since the system described is a single mixing junction, there is no topological description of this system. Consequently, the **node** is identified as a terminal node.

The methodology continues by applying IDENTIFY-POTENTIAL-HAZARD to the **node** description. IDENTIFY-POTENTIAL-HAZARD begins by applying FIND-ALL-PATHWAYS to the chemical species set of the **node**. This is obtained using the access functions

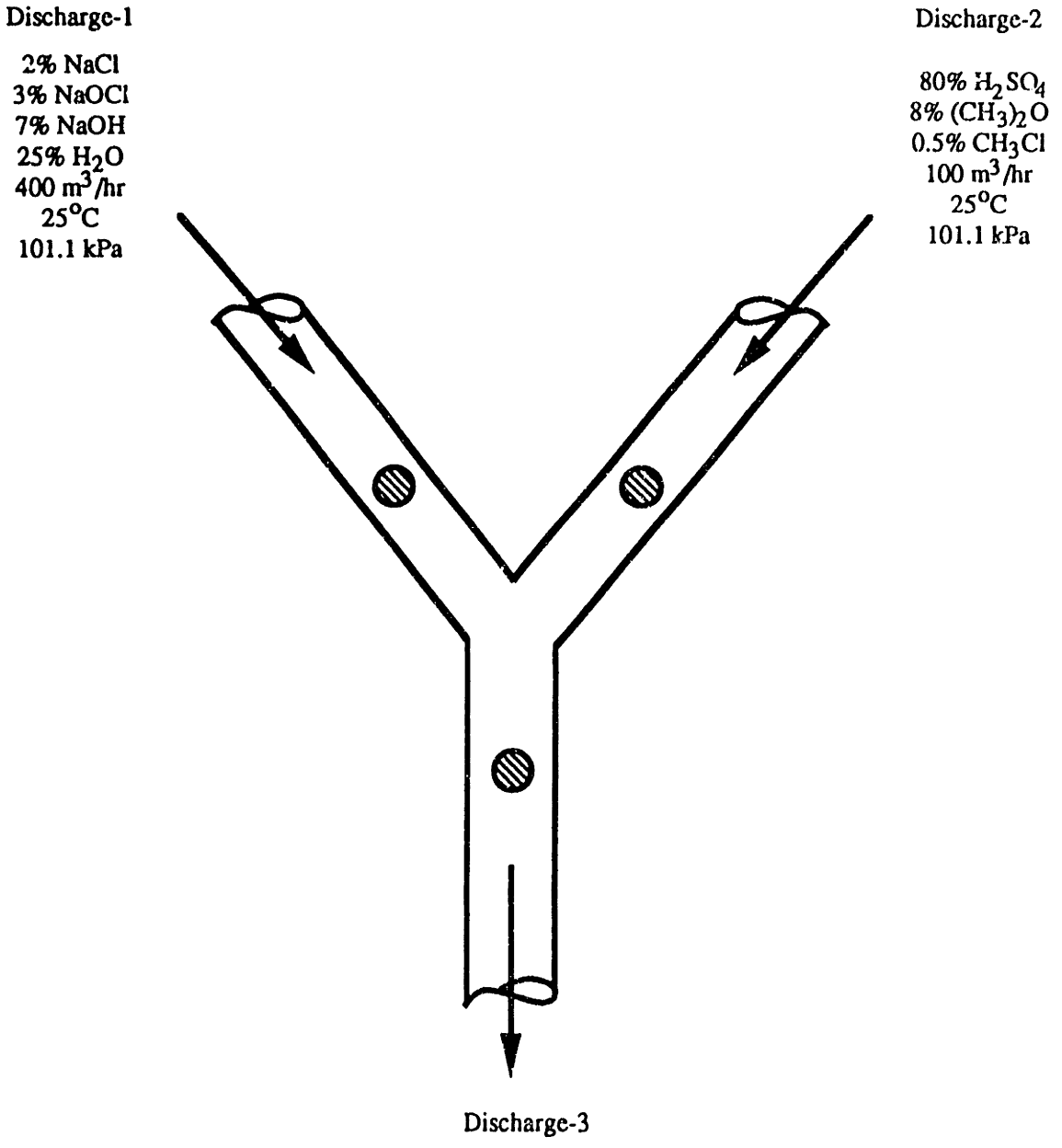


Figure 3-17: Discharge network

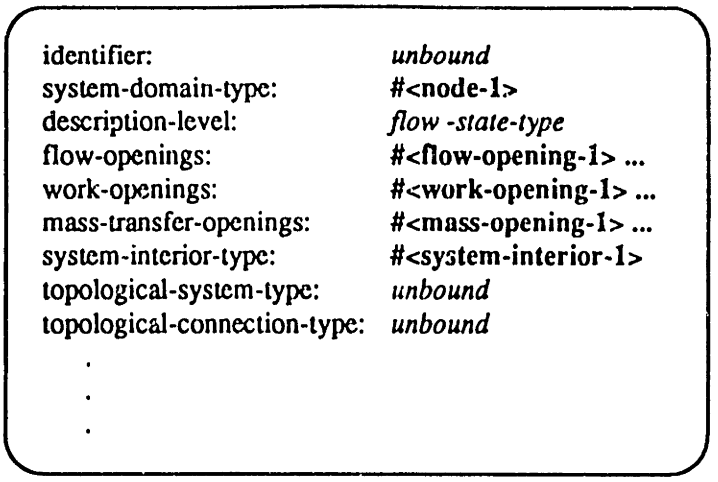


Figure 3-18: Node description: mixing junction

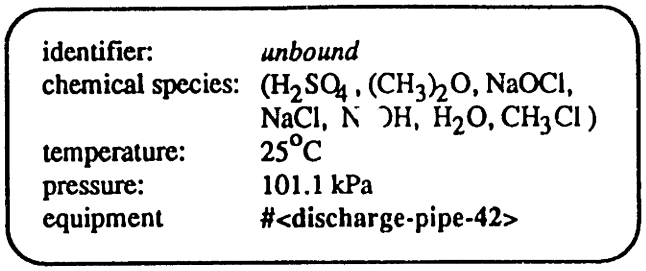


Figure 3-19: System description: mixing junction

provided by the process modeling language and is retrieved through the system-interior description (see Figure 3-19).

FIND-ALL-PATHWAYS begins by evaluating each species in the chemical species set individually to assess the inherent reactivity. Recall that the initialization of those compounds is already identified in relative chemical character; this was established when the modeling element, **chemical-behavior**, received its value(s) during the initialization process. Having evaluated each chemical species found in the chemical species set individually, FIND-ALL-PATHWAYS conducts a global routine which evaluates the various combinations of those species contained in the chemical species set. The call to FIND-ALL-PATHWAYS is given below:

(FIND-ALL-PATHWAYS :*substrates* chemical-species-set
:*operators* **K**
:*override-environment* **reaction-environment-1**)

FIND-ALL-PATHWAYS begins by calling the composite operators that comprise the domain of known operations on the substrates specified by the keyword argument :*substrates*. These operators loop over the substrate list and evaluate properties specified by **K**.

Reactions occuring in the chemical species set are initiated when FEASIBLE-P, a compound predicate method of **K** is evaluated on the sites identified by $K_{get\text{-}sites}$. These sites are passed to FEASIBLE-P which searches the **reaction-environment** to establish potential chemical transformations. Key reactions resulting from this evaluation are shown in Figure 3-20.

$$H_2SO_4 + NaOH \rightarrow NaHSO_4 + H_2O$$

$$H_2SO_4 + 2\ NaOH \rightarrow Na_2SO_4 + 2\ H_2O$$

$$H_2SO_4 + NaOCl \rightarrow HOCL + NaHSO_4$$

$$H_2SO_4 + 2\ NaOCl \rightarrow 2\ HOCL + Na_2SO_4$$

$$2\ NaOH + Cl_2 \leftrightarrow NaCl + NaOCl + H_2O$$

$$NaOCl + NaOH + 2\ H_2SO_4 \leftrightarrow 2\ NaHSO_4 + H_2O + Cl_2$$

$$NaOH + CH_3Cl \rightarrow CH_3OH + NaCl$$

$$CH_3OH + NaOH \rightarrow CH_3ONa + H_2O$$

$$NaOCH_3 + CH_3Cl \rightarrow CH_3\text{-}O\text{-}CH_3 + NaCl$$

$$CH_3\text{-}O\text{-}CH_3 + H_2SO_4 \rightarrow 2\ CH_3OH$$

$$CH_3\text{-}O\text{-}CH_3 + H_2SO_4 \rightarrow CH_3Cl + CH_3OH$$

Figure 3-20: Potential Reaction Set 1

Most notably are the reactions:

1. $H_2SO_4 + NaOH \rightarrow NaHSO_4 + H_2O$

2. $H_2SO_4 + 2NaOH \rightarrow Na_2SO_4 + 2H_2O$

3. $H_2SO_4 + NaOCl \rightarrow HOCl + NaHSO_4$

4. $H_2SO_4 + 2NaOCl \rightarrow 2HOCl + Na_2SO_4$

5. $2HOCl \leftrightarrow Cl_2 + H_2O_2$

6. $2H_2SO_4 + NaOH + 2NaOCl \leftrightarrow 2NaHSO_4 + H_2O + Cl_2$

Having identified potential pathways emanating from the chemical species set, the IDENTIFY-POTENTIAL-HAZARD collects the unique chemical constituents from the reaction list shown in Figure 3-20 and constructs the **state** associated with the various reaction processes. One such **state** is shown in Figure 3-21: pictorially, the sequence of states is shown in Figure 3-22. Of particular interest is **state-3**, which identifies an external mass and energy source corresponding to the addition of air and sunlight. Evaluation of this **state**, by FIND-ALL-PATHWAYS, identifies several additional reactions of significance (see Figure 3-23):

(FIND-ALL-PATHWAYS :*substrates* chemical-species-set
:*operators* **K**
:*override-environment* **reaction-environment-2**)

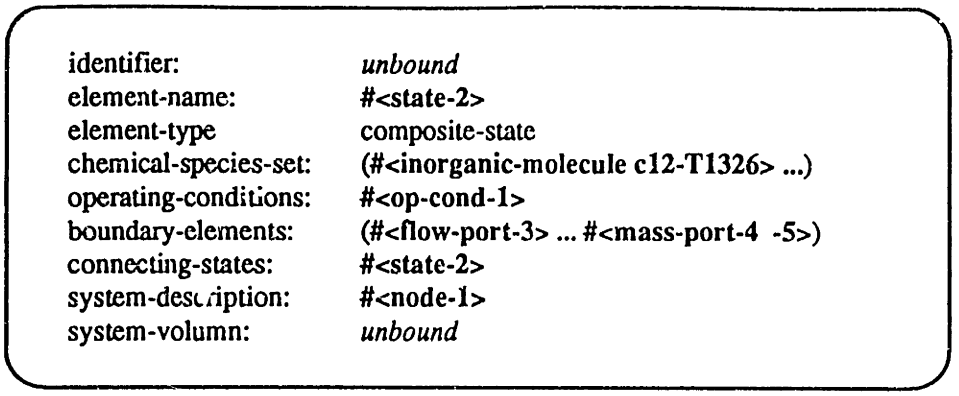


Figure 3-21: State description: gas phase

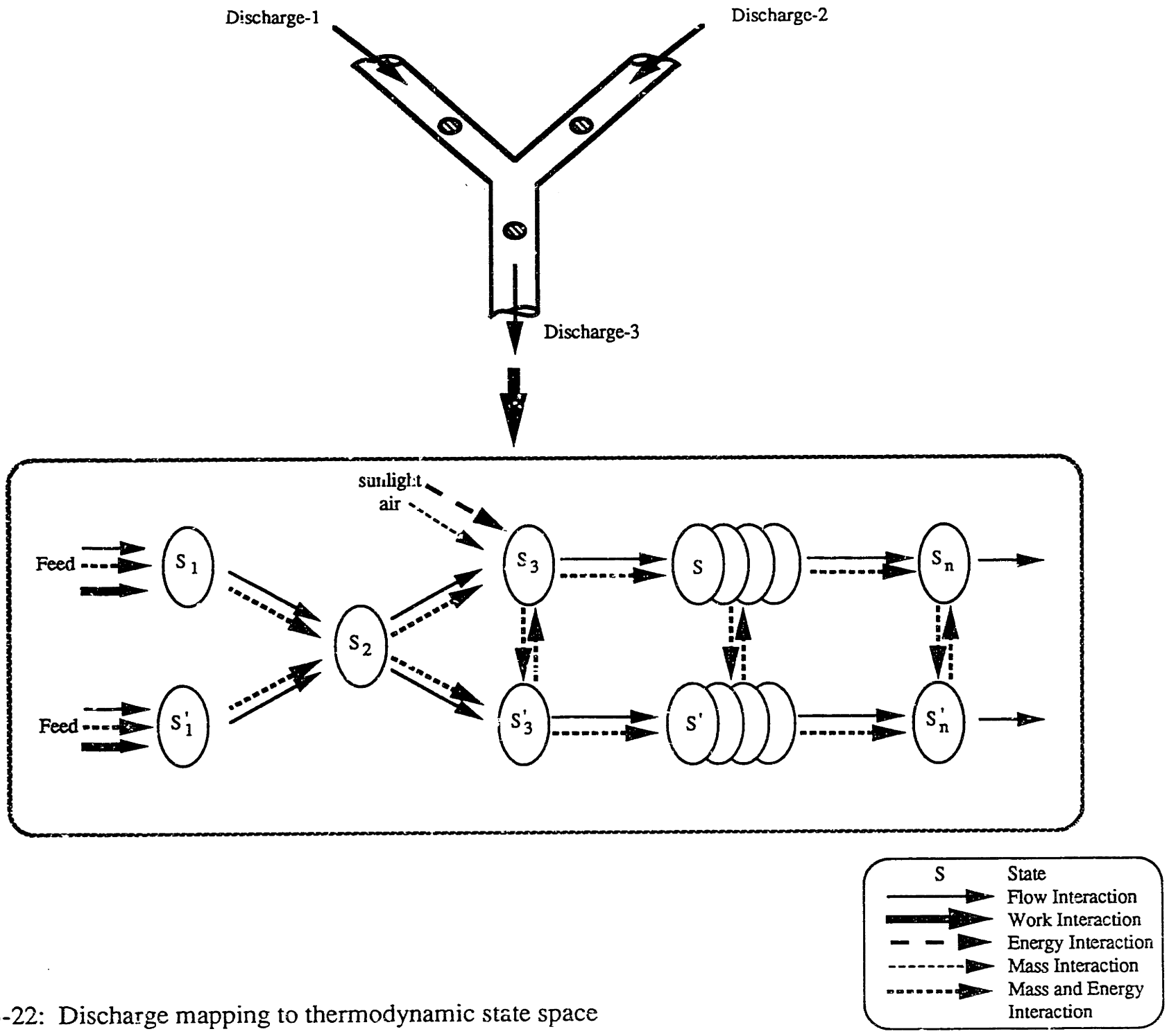


Figure 3-22: Discharge mapping to thermodynamic state space

$$2\ HOCl \leftrightarrow Cl_2 + H_2O_2$$

$$CH_3Cl + 3/2\ O_2 \rightarrow CO_2 + HCl + H_2O$$

$$CH_3\text{-}O\text{-}CH_3 + 3\ O_2 \rightarrow 2\ CO_2 + 3\ H_2O$$

$$Cl_2 \rightarrow 2\ Cl^{\cdot}$$

$$Cl_2 + CH_3\text{-}O\text{-}CH_3 \rightarrow HCl + CH_3\text{-}O\text{-}CH_2Cl$$

$$Cl_2 + CH_3Cl \rightarrow HCl + CH_2Cl_2$$

Figure 3-23: Potential Reaction Set 2

where,

reaction-environment-2

temperature: 298 K

wavelength: 500 nm

surface-type: non-metal

Using this new knowledge, **K** constructs a list of potential reactions. Key reactions resulting from this evaluation are:

1. $Cl_2 + CH_3\text{-}O\text{-}CH_3 \rightarrow HCl + CH_3\text{-}O\text{-}CH_2Cl$

2. $Cl_2 + CH_3Cl \rightarrow HCl + CH_2Cl_2$

3. $CH_3\text{-}O\text{-}CH_3 + 3O_2 \rightarrow 2CO_2 + 3H_2O$

4. $CH_3Cl + 3/2\ O_2 \rightarrow CO_2 + HCl + H_2O$

One chlorination pathway of dimethyl ether is shown in Figure 3-24; using the semantic relationships of CRL, the initiation reaction of a chlorination pathway can be accessed to identify the enabling mechanism(see Figure 3-25). This information is contained in the attribute enabling-conditions of the modeling element **reaction.** Expansion of this attribute value identifies the reaction conditions that enabled the disassociation. The methods embedded in CRL via the kinetic operators evaluate this information to determine the feasibility of different chemical processes.

By tracing the enabling conditions of these reactions and the sequence of preceding states, FIND-ENABLING-CRITERIA establishes the conditions necessary to enable the oxidation or chlorination of dimethyl ether and methyl chloride or both. FIND-

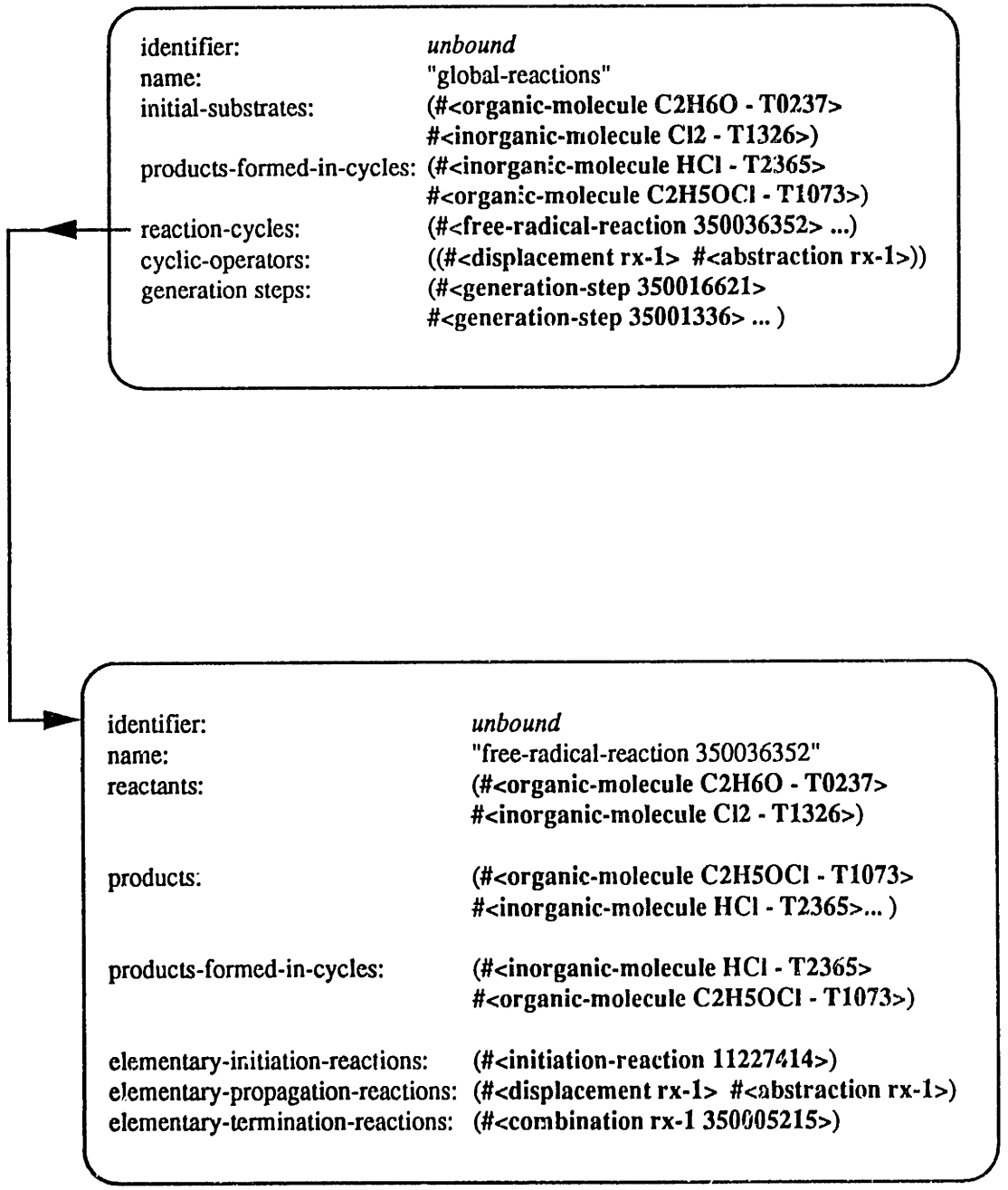


Figure 3-24: Free radical pathway generation

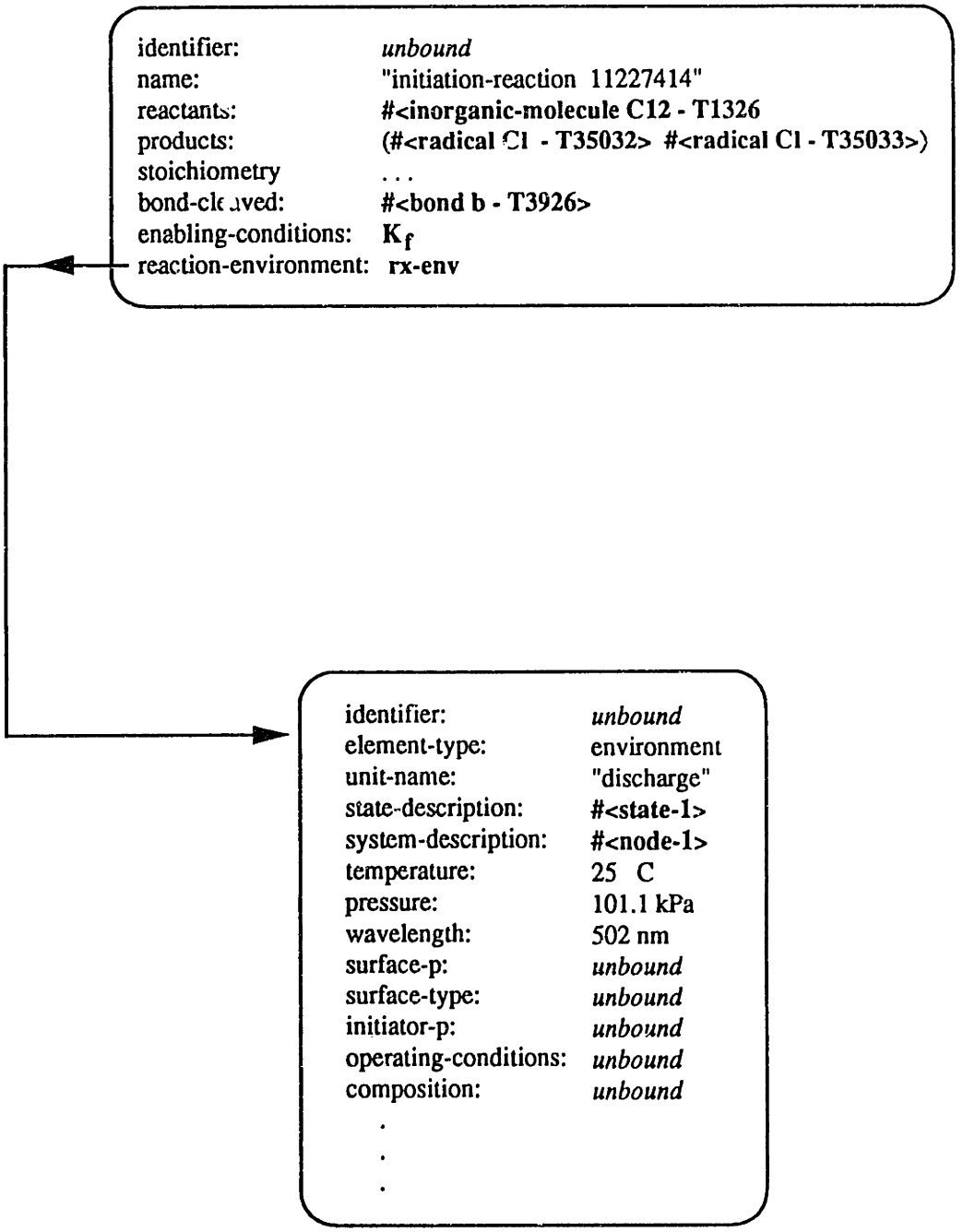


Figure 3-25: Initiation reaction expansion

ENABLING-CRITERIA establishes that the **mass-interaction-opening** connecting **state-2** and **state-3** establishes the mechanism by which chlorine is transported from the liquid phase to the gas phase; and, the external **mass-interaction-3** and **energy-interaction-3** of **state-3** enable the chlorination and oxidation reactions. FIND-ENABLING-CRITERIA, utilizes the utilities of the process modeling language to establish the transfer of dimethyl ether from the liquid phase (S_3') to the gas phase (S_3). These utilities were shown in Figure 3-6; they are referred to as handlers. The unique states identified in Figure 3-22 are denoted S_1, S_2, S_3, S_3', S_n, and S_n'.

Using the utilities of the process modeling language, the IDENTIFY-POTENTIAL-HAZARD methodology applies EVALUATE-PHYSICAL-REACTIONS to the unique state descriptions. These utilities assess the feasibility of various physical reactions, such as overpressurizing or overheating, and when such a condition exists, IDENTIFY-POTENTIAL-HAZARD appends the reaction onto the potential-reaction-list. Each **reaction** of the potential-reaction-list is then analyzed to find the enabling criteria.

Consider, for example, identification of the enabling criteria for the chlorination of dimethyl ether which is associated with **state-3**. The routine begins by first identifying the enabling conditions of the reaction itself as is shown in Figure 3-25. These conditions establish the criteria necessary for chlorine disassociation. Similarly, by tracing the genesis of the reactants, chlorine and dimethyl ether, to **state-3** and using the attribute connecting-states of **state** we can construct a complete history of the events and conditions which ultimately led to **state 3**.

An analysis is conducted to evaluate potentially hazardous pathways of this reaction using IDENTIFY-NONDISSIPATED-PATHWAYS. This method relies on the availability of the utilities associated with the underlying process modeling language. With these utilities the vapor phase availability of dimethyl ether is assessed to be 24 kmol/hour and

that of methyl chloride to be 0.09 kmol/hour. By combining this information with the knowledge garnered from the reactions involving dimethyl ether and methyl chloride, an evaluation of the pressure rise associated with these various reaction sequences can be evaluated (see Figure 3-26). Assessment of this information in the context of the node description suggests that potential hazards involving methyl chloride can be removed from the potential-hazard-list since the **node** is capable of dissipating its effects.

Using the information embedded in the reactions of interest and the associated states, and by tracing the enabling conditions of these states, the logical event sequence can be constructed for the various reactions. Figure 3-27 shows the event sequence for the oxidation of dimethyl ether. Notice that the probability of ignition is left undefined and is presumed to be obtained from an external data bank. Similarly, Figure 3-28 shows the event sequence for the chlorination of dimethyl ether. Therein, the probability of the initiation mechanism is defined by the availability of the light source; this is established in the node description. Alternatively, the chlorination reaction may be the initiator source. This leads to the hazard event sequence shown in Figure 3-29.

3.4.2 Case Study 2: solvent selection

We now consider the selection of a solvent for the storage of t-butyl hydroperoxide (thp). The process flowsheet consists of a single vessel. Application of GLOBAL-HAZARD-IDENTIFICATION to this flowsheet begins by applying MAKE-PROCESS-TRANSFORMATION to the process flowsheet. MAKE-PROCESS-TRANSFORMATION returns a single **node**; this **node** is a terminal node. The node representation of this storage vessel contains no flow, heat, work, or mass openings and no topological connectors. As a consequence, the evaluation of potential hazards reduces to the analysis of the system interior description. This is accomplished by applying IDENTIFY-POTENTIAL-HAZARD to the **node** description.

Vapor Phase Availability

$(CH_3)_2O$	~ 24.0 kmol/hr
CH_3Cl	~ 0.09 kmol/hr

Reaction Sequence	T_{auto} (°K)	P_{rise} (atm)
$(CH_3)_2O$/Air	623	~ 12
CH_3Cl /Air	930	~ 1
$(CH_3)_2$/Cl	_	~ 10

Figure 3-26: Vapor phase availability and potential consequences

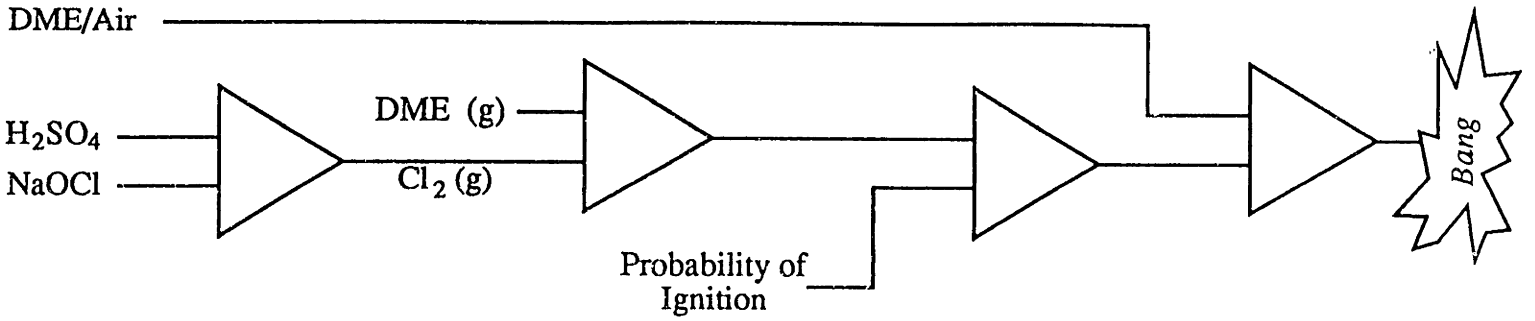


Figure 3-27: Dimethyl ether oxidation: hazard event sequence

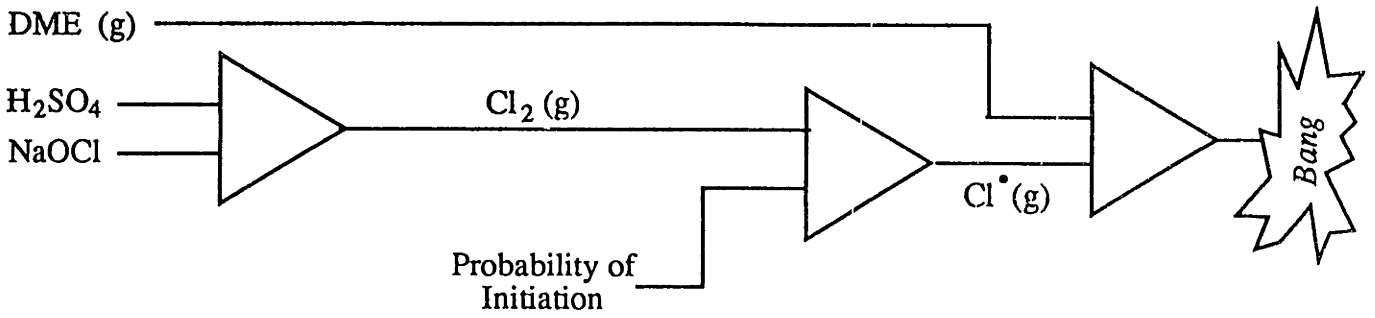


Figure 3-28: Dimethyl ether chlorination: hazard event sequence

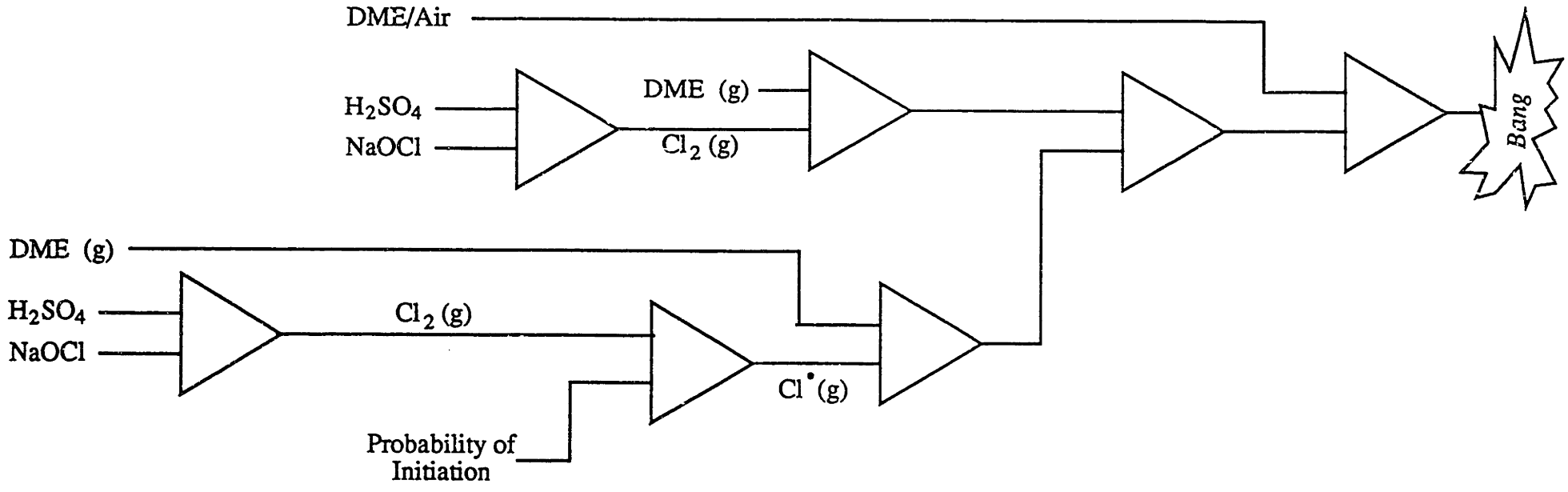


Figure 3-29: Dimethyl ether chlorination: hazard event sequence

The IDENTIFY-POTENTIAL-HAZARD routine applies FIND-ALL-PATHWAYS to the chemical species set associated with the **node**. This method begins by evaluating the inherent reactivity of thp in the presence of its stabilizer. It does this by restricting the substrates passed to FIND-ALL-PATHWAYS:

(FIND-ALL-PATHWAYS :*substrates* **(t-butyl hydroperoxide sulfuric acid)**
:*operators* **K***
:*override-environment* **reaction-environment-1**)

where,

reaction-environment-1

temperature: 298 K

wavelength: nil

surface-type: non-metal

and **K*** is defined as the set of feasible composite operations with free energy changes greater than 10 kcal/mol.

Several reactions identified are shown in Figure 3-30. Most notable are the protonation reactions, initiated by the stabilizer, leading to the formation of such products as t-butyl ether, oxygen, t-butyl alcohol and isobutylene; as well as the free radical reactions of thp which lead to the formation of acetone, oxygen, and various alcohols and hydrocarbons.

Figure 3-31 shows several of the free radical pathways for the disassociation of thp. Notice that the attribute values of **free-radical-reactions** not only identifies the reactants from the initial substrate list but also the products, elementary reactions, and composite operations involved in establishing the propagation cycles. The values of these attributes are particularly important because once an initiation mechanism is established, the propagation cycle may be turned numerous times. Oftentimes, in the case of auto-

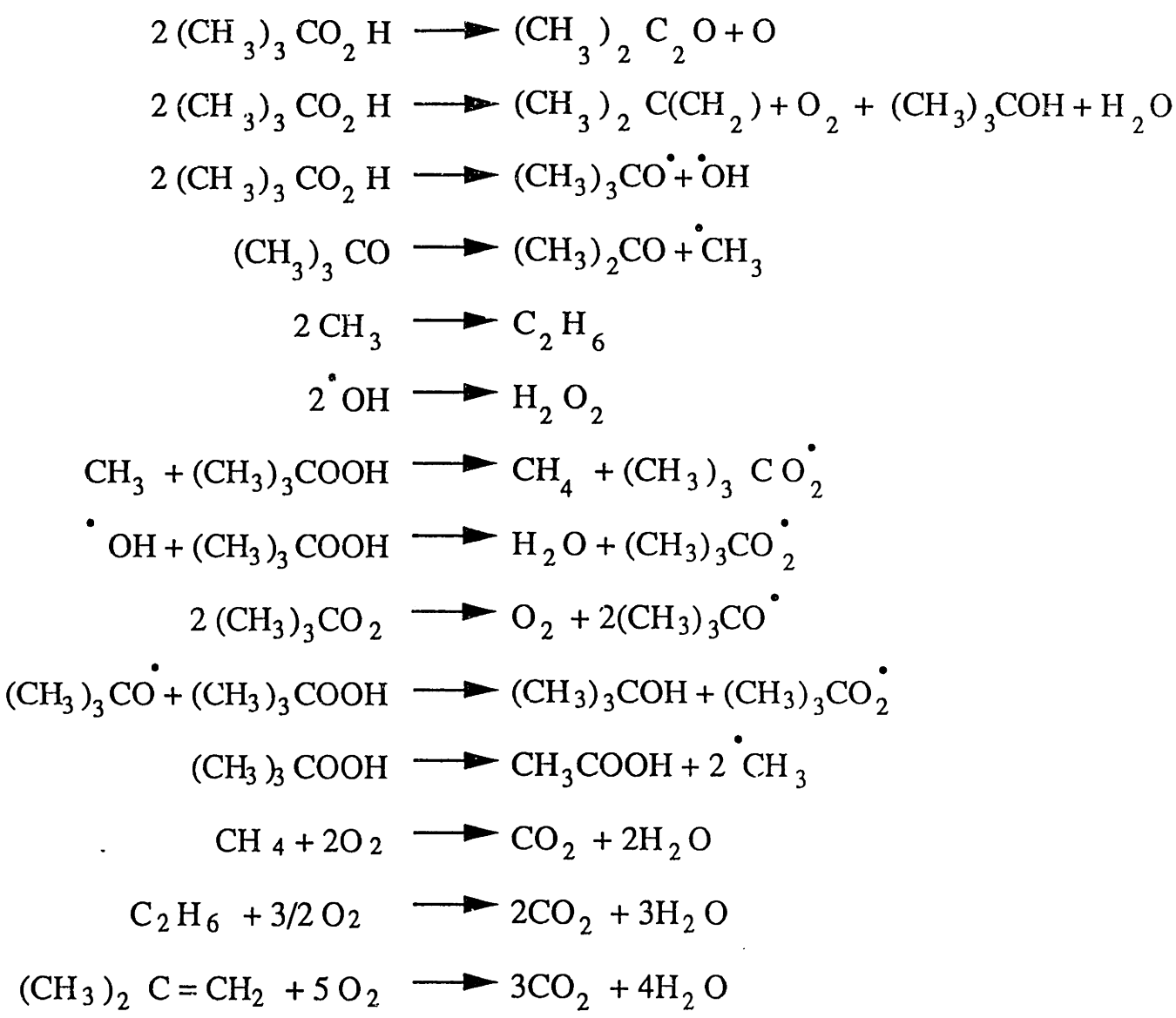


Figure 3-30: Reactions of t-butyl hydroperoxide

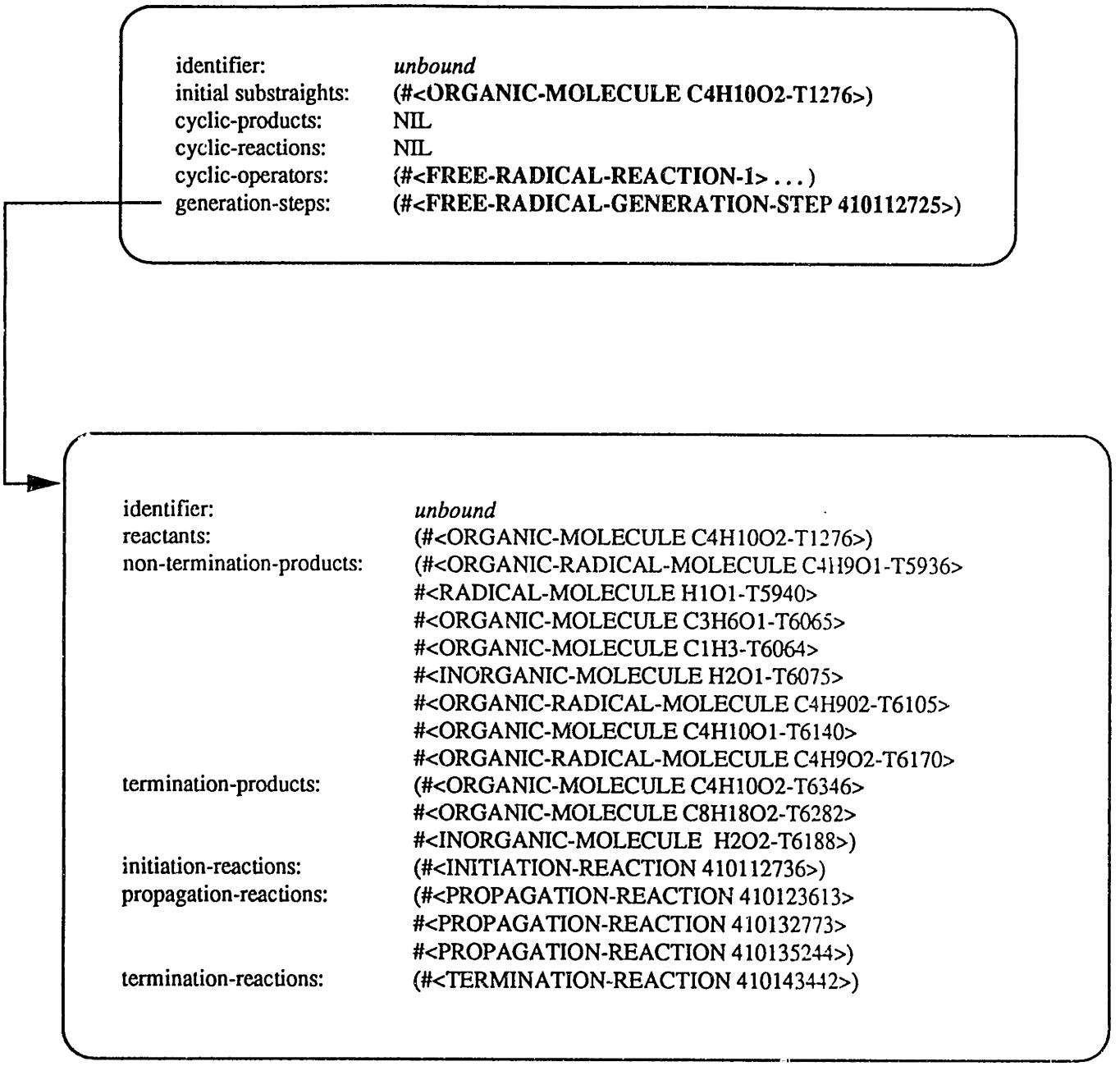


Figure 3-31: Description: free-radical-reactions with generation step expansion

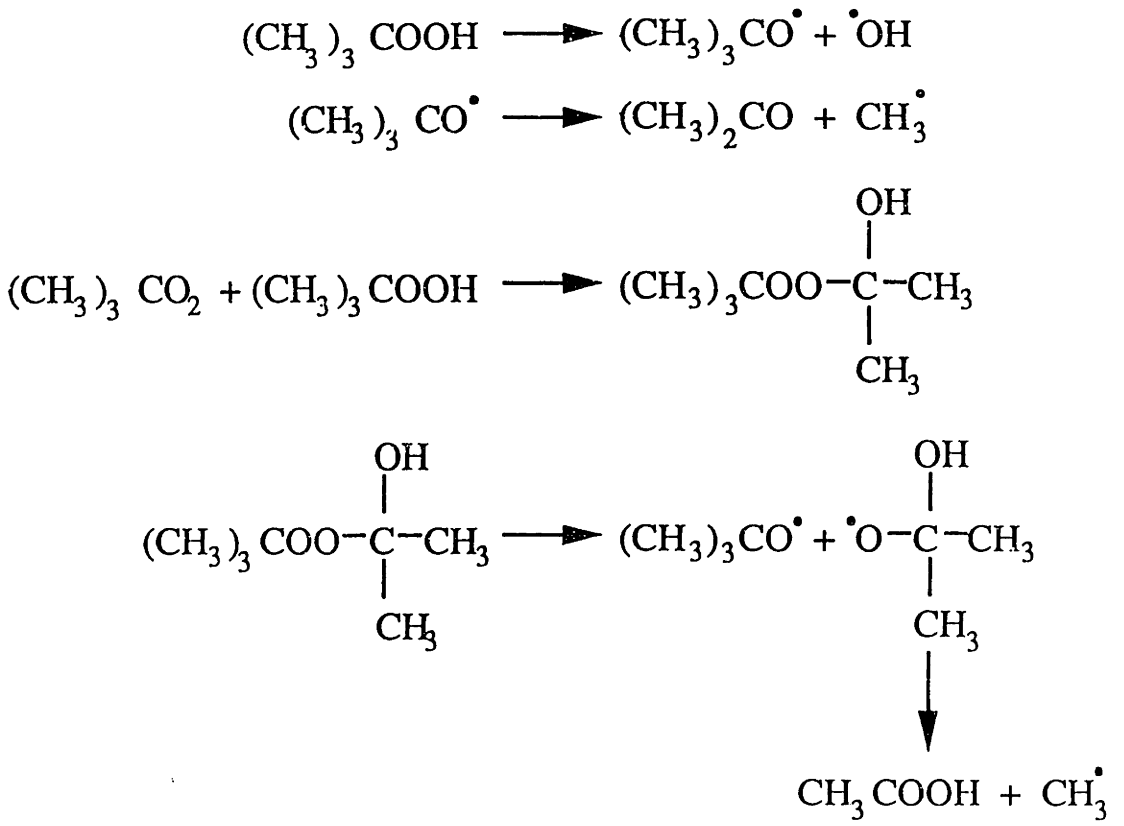
oxidation and chlorinations, the cycle may be turned several hundred to several thousand times for each initiation. Because of the importance of initiation mechanisms in free radical pathways, FIND-ALL-PATHWAYS gives the user the utility to postulate the presence of initiator albeit, unknown at the time, by setting the value of the keyword *:initiator-p* to true. Figure 3-32 describes some for the information contained in the elementary reactions which comprise the **free-radical-reaction.**

Having established the potential set of reactions which can involve thp, unique chemical constituents, whether they be reactant or product, can then be collected. This extended chemical species set must be used to evaluate the potential hazards associated with various solvents. For example, if the choice of solvent in which thp is to be stored includes t-butanol, propanol, ethanol, and methanol these species must be evaluated independently by FIND-ALL-PATHWAYS and then included in the extended chemical species set and reevaluated.

The reevaluation of the extended chemical species set with potential solvent leads to a particularly interesting reaction: the reaction between acetic acid and the solvent forming the respective alkene. Although we have shown in Figure 3-30 that the isobutylene can potentially be formed by several mechanisms, we now have identified a free radical reaction which forms the necessary reactant, acetic acid, in the propagation cycle. This is particularly important because there is no longer a one to one correspondence between the acetic acid produced and the initiation mechanism. This occurs because thp is being consumed to form acetic acid (see Fig. 3-33). More importantly, an elimination reaction between acetic acid and the potential solvents can lead to the generation of their respective alkenes. As a consequence, the solvent and the stored material, thp, can be consumed in a hybrid cycle, forming the alkene, once thp dissociates.

name:	"radical-combination/coupling"
reactants:	(#<RADICAL-MOLECULE H1O1-T5940> #<RADICAL-MOLECULE H1O1-T5940 >)
products:	(#<INORGANIC-MOLECULE H2O2-T6188>)
substrate 1:	#<RADICAL-MOLECULE H1O1-T5940>
substrate 2:	#<RADICAL-MOLECULE H1O1-T5940>
substrate 1-atoms:	(#<ATOM O-T5937> #<ATOM H-T5938>)
substrate 1-bonds:	(#<BOND b-T5939>)
substrate 2-atoms:	(#<ATOM O-T5937> #<ATOM H-T5938>)
substrate 2-bonds:	(#<BOND b-T5939>)
reactions:	(#<MICRO-COMBINATION 410142040>)
micro-products:	(#<INORGANIC-MOLECULE H2O2-T6188>)
micro-stoichiometry:	((#<RADICAL-MOLECULE H1O1-T5940> . -1) (#<RADICAL-MOLECULE H1O1-T6174> . -1) (#<INORGANIC-MOLECULE H2O2-T6188> . +1)
micro-rx-heat:	(-34.894836 . "kcal/mol")
literature-reference:	unbound
enabling-conditions:	#<K_t>
.	.
.	.
.	.

Figure 3-32: Description: combination reaction



Reaction of acetic acid with solvent:

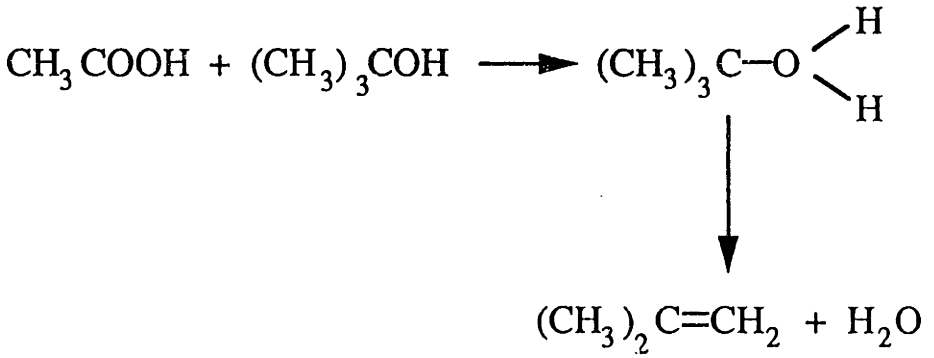


Figure 3-33: Solvent reactions of t-butyl hydroperoxide decomposition

An appended reaction list resulting from these evaluations is shown in Figure 3-34. Notice that the reaction between the solvent and acetic acid forming the respective alkene can only be formed tertiary, secondary, and primary substituted carbon solvents. This is a necessary precondition for a first order elimination reaction. The enabling conditions, contained in the composite operator, establishes the vehicle for preventing a first order elimination reaction between acetic acid and select solvents.

Using this information, the EVALUATE-STATES generates the **state** associated with each of these reactions. Unique states contained in associated-states are then passed to EVALUATE-PHYSICAL-REACTIONS. This method assesses the likelihood of overpressurizing the storage vessel or overheating the contents of that vessel; evaluation of this method is assisted by the utilities provided in the underlying process modeling language. Specifically, the multiphase component handler of this package allows vapor phase availability of these difference constituents to be assessed. EVALUATE-PHYSICAL-REACTIONS then appends possible physical reactions to the potential-reaction-list.

Each of the reactions in the potential reaction list is then processed using FIND-ENABLING-CRITERIA to establish enabling conditions for the reaction. Since CRL associates enabling conditions with each reaction, using the information embedded in the composite operators, this task is straightforward requiring only the collection of information contained in that attribute. Access to modeling element attribute values is provided by the semantic relationships of CRL. This is shown in Figure 3-32 for the disassociation of thp. Because each reaction shown is associated with elements contained in the storage vessel, potentially dissipative pathways must rely on containment of the overpressuring condition arising from a given reaction sequence. Consequently, the reactions of importance (e.g. overpressurizing resulting from the combustion of alkenes or the liberation of gaseous products from the fluid) can be identified immediately as

$$2\,(CH_3)_3CO_2H \longrightarrow (CH_3)_2C_2O + O$$

$$2\,(CH_3)_3CO_2H \longrightarrow (CH_3)_2C(CH_2) + O_2 + (CH_3)_3COH + H_2O$$

$$2\,(CH_3)_3CO_2H \longrightarrow (CH_3)_3CO^{\bullet} + {}^{\bullet}OH$$

$$(CH_3)_3CO \longrightarrow (CH_3)_2CO + {}^{\bullet}CH_3$$

$$2\,CH_3 \longrightarrow C_2H_6$$

$$2\,{}^{\bullet}OH \longrightarrow H_2O_2$$

$$CH_3 + (CH_3)_3COOH \longrightarrow CH_4 + (CH_3)_3CO_2^{\bullet}$$

$${}^{\bullet}OH + (CH_3)_3COOH \longrightarrow H_2O + (CH_3)_3CO_2^{\bullet}$$

$$2\,(CH_3)_3CO_2 \longrightarrow O_2 + 2(CH_3)_3CO^{\bullet}$$

$$(CH_3)_3CO^{\bullet} + (CH_3)_3COOH \longrightarrow (CH_3)_3COH + (CH_3)_3CO_2^{\bullet}$$

$$(CH_3)_3COOH \longrightarrow CH_3COOH + 2\,{}^{\bullet}CH_3$$

$$CH_4 + 2O_2 \longrightarrow CO_2 + 2H_2O$$

$$C_2H_6 + 3/2\,O_2 \longrightarrow 2CO_2 + 3H_2O$$

$$(CH_3)_2C=CH_2 + 5\,O_2 \longrightarrow 3CO_2 + 4H_2O$$

$$CH_3CO_2H + (CH_3)_3COH \longrightarrow (CH_3)_2C=CH_2 + H_2O + \text{acid}$$

$$CH_3CO_2H + CH_3CH_2CH_2OH \longrightarrow (CH_3)CHCH_2 + H_2O + \text{acid}$$

$$CH_3CO_2H + CH_3CH_2OH \longrightarrow CH_2=CH_2 + H_2O + \text{acid}$$

$$CH_3CO_2H + CH_3OH \longrightarrow \text{no reaction}$$

Figure 3-34: Reactions of t-butyl hydroperoxide and select solvents

potential hazards, with their enabling conditions, when criteria for dissipation are not achieved.

Thus, while the decomposition of thp is well established [Walling, 1957, de Groot and Hupkens van der Elst, 1981], the pathways described suggest that if means are found for initiating the decomposition of thp (e.g. light, heat, catalyst), decomposition products may lead to the formation of acetic acid which in turn can react with common solvents (e.g. t-butanol, isopropanol, and to a lessor extent ethanol) forming potentially volatile and ignitable products (e.g. isobutylene). Consequently the intermingling of these systems may lead to an overpressurized condition in the storage vessel; particularly when the storage vessel is not vented. Results of this analysis show, however, that the formation of the respective alkenes can be minimized through an appropriate choice of solvent. Specifically, methanol provides greater inherent safety than ethanol which is followed by propanol and lastly by t-butyl alcohol.

3.4.3 Case Study 3: aniline production

The ability to elucidate competing pathways and determine when change of control occurs is of particular importance in hazard identification. An understanding of the relationship between reaction pathway topography and design and operating characteristics allows us to mitigate or constrain the reaction trajectory. Consider for example, the catalytic production of aniline from nitrobenzene and hydrogen. Limiting our attention to the reactor, the initialization routine of GLOBAL-HAZARD-IDENTIFICATION transforms the reactor into a single terminal process node. The identification of hazards within this **node** are identified by applying IDENTIFY-POTENTIAL-HAZARD to the **node.**

IDENTIFY-POTENTIAL-HAZARD begins by applying FIND-ALL-PATHWAYS to the chemical species set. This set, which is bound to the **node,** is shown in Figure 3-35. The call to

FIND-ALL-PATHWAYS and its application to the chemical species set is shown below for composite operators involving hydrogenation:

(FIND-ALL-PATHWAYS :*substrates* CSS :*operators* $\mathbf{K_{hydrogenation}}$)

Key reactions identified by FIND-ALL-PATHWAYS are:

1. $\varphi\text{-}NO_2 + H_2 \rightarrow \varphi\text{-}NO + H_2O$

2. $\varphi\text{-}NO + H_2 \rightarrow \varphi\text{-}NHOH$

3. $\varphi\text{-}NHOH + H_2 \rightarrow \varphi\text{-}NH_2 + H_2O$

These reactions combine forming the overall reaction:

$$\varphi\text{-}NO_2 + 3H_2 \rightarrow \varphi\text{-}NH_2 + 2H_2O$$

A less restrictive evaluation of FIND-ALL-PATHWAYS on a chemical species set (e.g. calling FIND-ALL-PATHWAYS with :*operators* **K** or $\mathbf{K^*}$) results in several additional reactions of interest. Most notably, is the decomposition reaction of nitrobenzene and the disproportionation of nitrobenzene with N-phenyl hydroxylamine forming aniline and nitrobenzene:

1. $\varphi\text{-}NO_2 \rightarrow$ products

2. $\varphi\text{-}NO + \varphi\text{-}NHOH \rightarrow \varphi\text{-}NH_2 + \varphi\text{-}NO_2$

Several of the pathways which emanate from these reactions and which are constructed by FIND-ALL-PATHWAYS are shown in Figure 3-36. Figure 3-37 shows how this information is managed by CRL using modeling elements. This information is unavailable in conventional chemical synthesis programs (e.g. SECS and CYCLOPS)

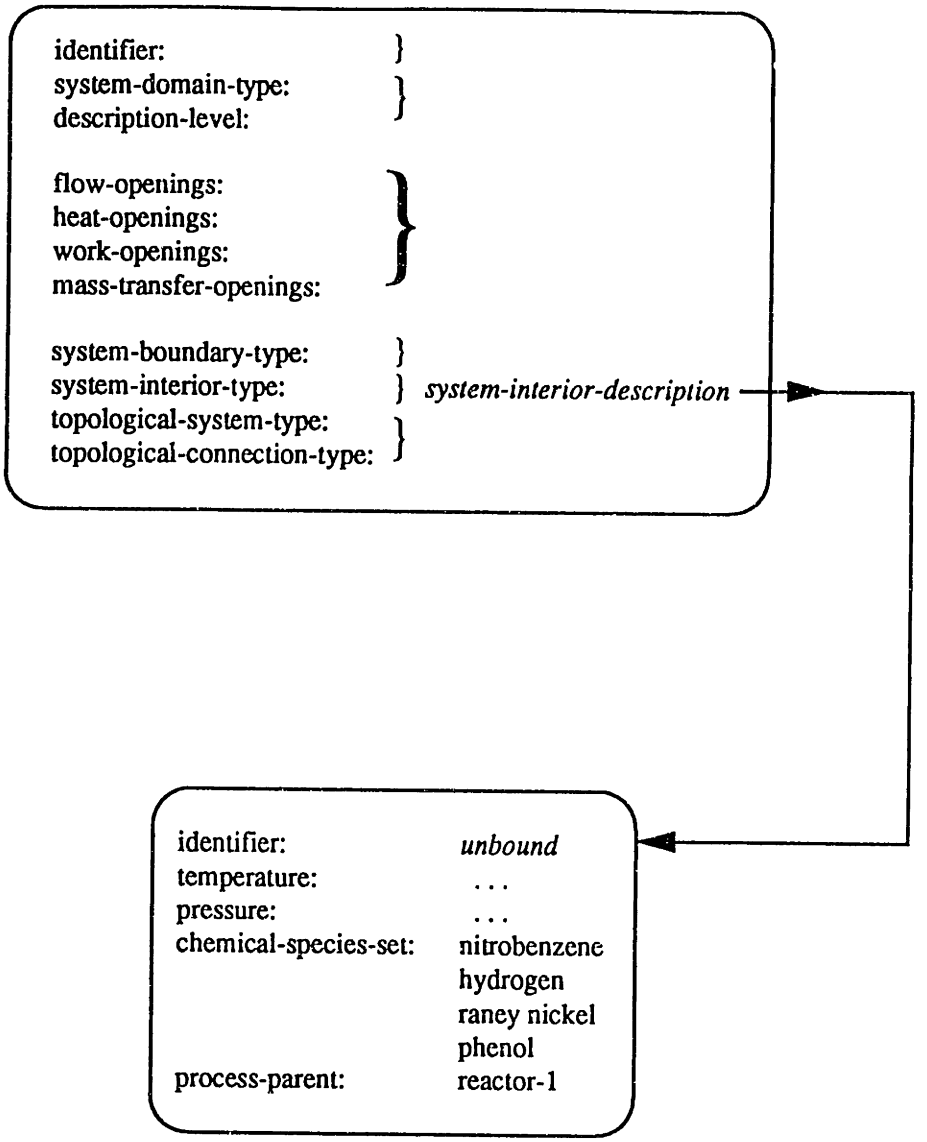


Figure 3-35: Node description

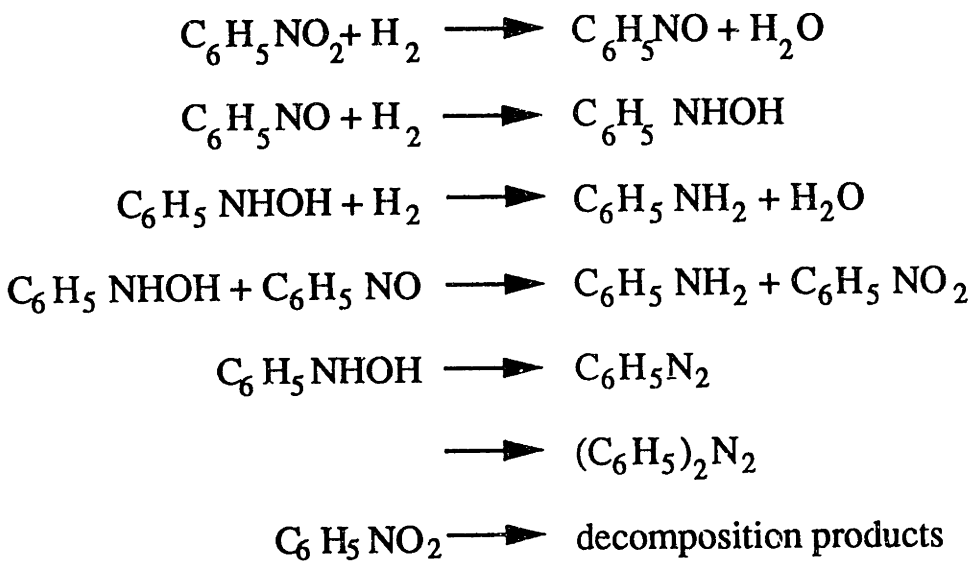


Figure 3-36: Potential reactions of nitrobenzene

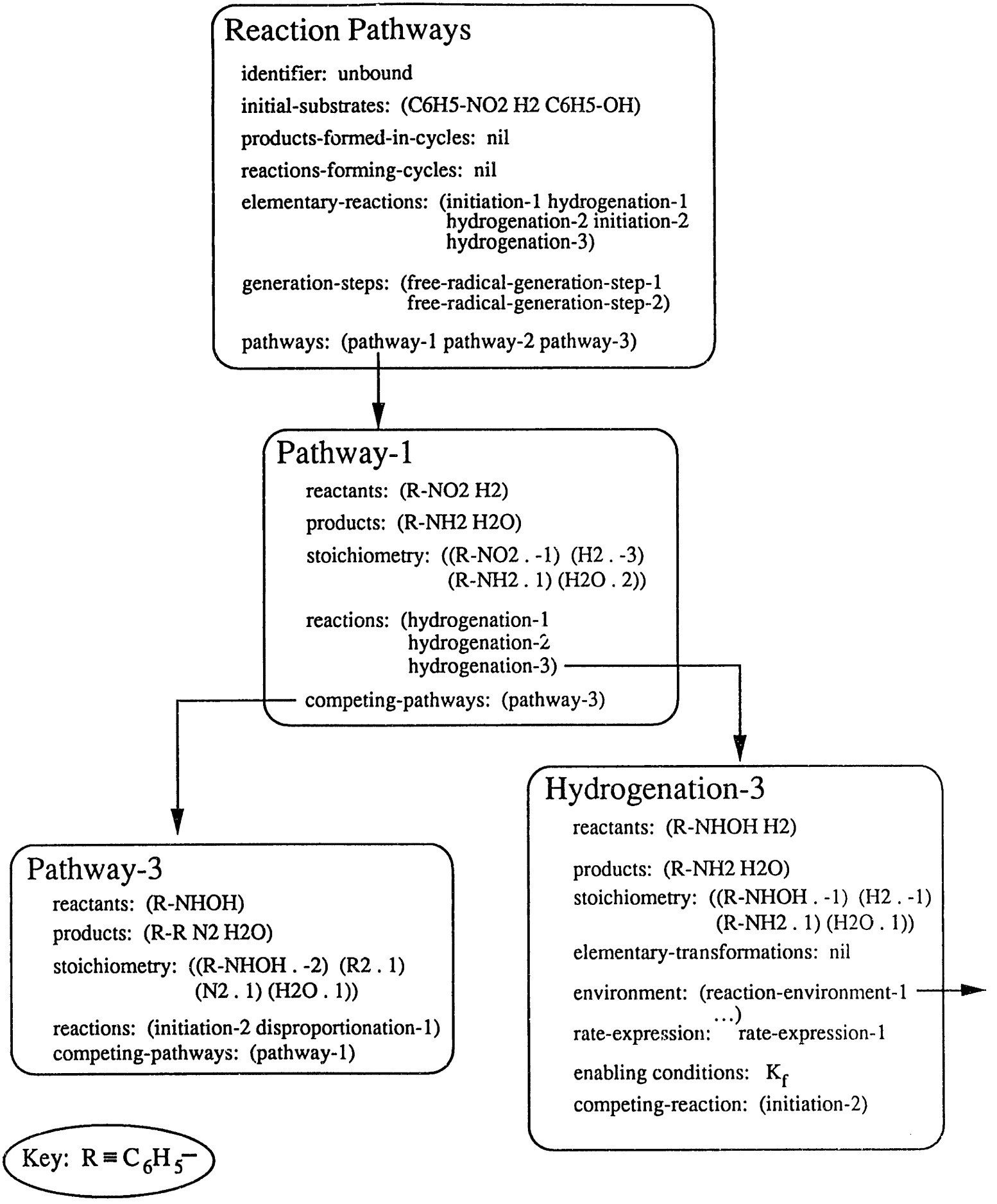


Figure 3-37a: Reaction elements and their linkage

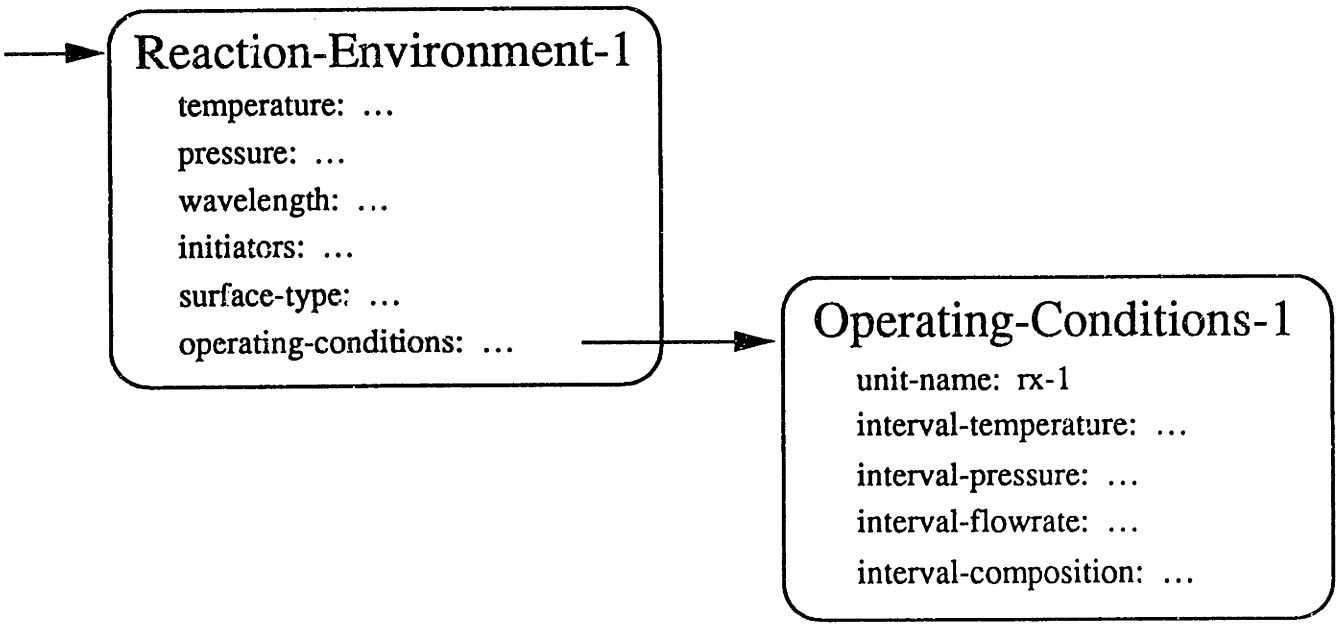


Figure 3-37b: Reaction modeling elements and their linkage

and provides an explanation as to why the reaction and the intermediates formed using the semantic relationships advanced by CRL.

For example, **pathway-1** denoting the transformation of nitrobenzene into aniline knows that it ***is-disaggregrated-in*** three separate individual reactions denoted **hydrogenation-1**, **hydrogenation-2**, and **hydrogenation-3**. Similarly, it knows that it ***is-abstracting*** the more general global reaction, **reaction-pathways**. It also understands which pathways are in competition; those in turn can be abstracted or disaggregrated using the semantic relationships of CRL. In this way, the chemical sequence of events which led to the reaction can be traced all the way back to the reaction conditions which enabled it; and these can be related to the operating conditions of the processing unit using the parent-equipment attribute of system-description (an attribute of **node**). This is the mechanism by which FIND-ENABLING-CRITERIA traces out the sequence of events which lead to the reaction of interest. Figure 3-38 shows the path which leads to the formation of aniline.

Because of the information contained in the objects that comprise this path, we know precisely the conditions which led to the creation of each element contained in the path. For example, by accessing the heat-of-formation attribute associated with the modeling element **reaction**, we identify that heat liberation is non-uniform during hydrogenation. This implies that an accumulation of the intermediate ϕ-NHOH could lead to an uncontrollable energy release.

More importantly, the tree spanning generation pathways for aniline production (see Fig. 3-38) makes explicit that hydrogen injection is an insufficient means for controlling reaction temperature. This results because of the disproportionation reaction identified by FIND-ALL-PATHWAYS. As a consequence, moderation of the hydrogen injection rate and the cooling fluid rate are necessary to control the heat of reaction. The identification that the heat of reaction cannot be controlled through hydrogen addition alone is an

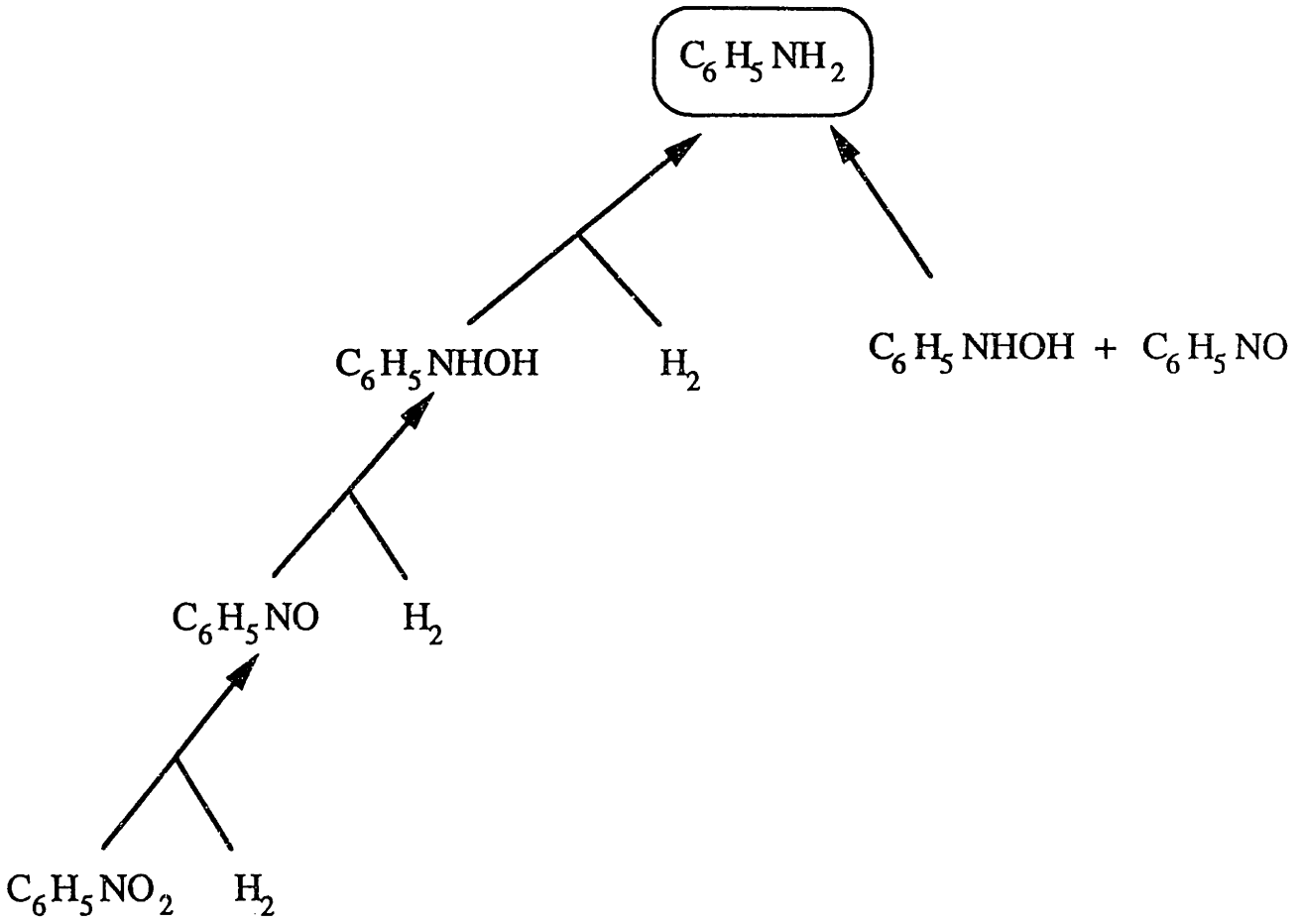


Figure 3-38: Reaction path to aniline

important parameter for safe reactor operation because the experimental detection of the N-phenyl hydroxylamine disproportionation with nitrosobenzene is difficult without prior knowledge of its occurrence [Stoessel, 1989].

Since CRL specifies the preconditions associated with each reaction, the reaction conditions which enabled it can be identified. This allows us to make explicit such information as: the decomposition temperature of φ-NO_2; the disproportionation temperature of φ-NHOH; and, the preconditions for the exothermic formation of such products as azobenzene and diazobenzene. By expanding the node description to include the atmosphere around the reactor (e.g. air), FIND-ALL-PATHWAYS would have identified combustion reactions involving hydrogen, phenol, and raney nickel as well.

As a consequence of this analysis, we can see that if one started with only an understanding of the overall reaction, that is, the hydrogenation of nitrobenzene to form aniline and water, the analyst may not have recognized the need to control reaction temperature while moderating both cooling rates and hydrogen injection rates. Lacking this information, the loss of recirculation may not have been understood nor the need to provide emergency cooling to satisfy additional duty requirements resulting from N-phenyl hydroxylamine accumulation or potentially the need for redundant recirculation pumps.

3.5. Conclusions

A new approach for the identification of hazards has been presented. The approach begins with the inductive determination of hazardous reactions and proceeds deductively through the network of processing steps to identify completely all sources of initiation. The methodology uses extensively specialized modeling languages for the generation of chemical reactions and the analysis of those reactions within the processing environment. These languages help to remove the representational constraint imposed by past

approaches. The methodology is formal. It unifies the identification of potential hazards and the pathways leading to hazards at all stages of the design process in a systematic manner. Because of the approach and the utility of the specialized modeling languages, the methodology is capable of transferring information derived at one design phase to another. This enables automation of the methodology.

More importantly, the approach can guarantee completeness within the scope of the modeling effort which defines the process in an efficient manner. Implications of this approach have been compared with and contrasted to conventional methodologies; particularly close attention has been paid to issues concerning computational efficiency.

3.6. Bibliography

Abelson, Harold, Gerald Jay Sussman and Julie Sussman, Structure and Interpretation of Computer Programs, MIT Press, Cambridge, Massachusetts, 1985.

Atallah, S., "Assessing and Managing Industrial Risk", Chemical Engineering, September 8, 1980

Batstone R., International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 5.126

Battelle, Guidelines for Hazard Evaluation Procedures, AIChE Press, 1985

Boykin, R.F. and M. Kazarians, "Quantitative Risk Assessment for Chemical Operations", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.87

Carson and Mumford, "Analysis of Incidents Involving Major Hazards in the Chemical Industry", J. Haz. Mat., 1979, 3 p.149-165

Cormen, T. H., C. E. Leiserson, R. L. Rivest, "Introduction to Algorithms," M.I.T. Press, Cambridge, MA (1990)

Cox, R.A., "An Overview of Hazard Analysis", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.37

Culbertson, T.L. and A.H. Searson, "Exxon Facility Design Assessment and Control of Hazards", internal publication, September, 1983

Dale, S.E., "Cost Effective Design Considerations for Safer Chemical Plants", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, 1987, AIChE, p. 3.79

de Groot, J. J. and F. Hupkens van der Elst, "Thermal Properties of Peroxides", I. Chem. E. Symposium Series No. 68, 1981, p. 3/V:1

Haastrup, P., "Design Error in the Chemical Industry", IChemE Symposium Series, No. 80, 1983, p. J15

Henning, H., Leone, H., and Stephanopoulos, G, "MODEL.LA: A Modeling Language for Process Engineering, Part I: The Formal Framework," Comp. Chem Engng., Vol. 14, No. 8, p. 813 (1990)

Hoffmann, J.M., Chemical Process Hazard Review, Chemical Process Hazard Review, American Chemical Society, 1985, p. 1

J. B. Hendrikson, "Fragmentations and Rearrangements in Organic Synthesis," J. Am. Chem. Soc., Vol. 108, pp. 6748-6756, 1986

Kritikos, Theodoros, "A Model for Process Design Automation", Doctoral Thesis, Massachusetts Institute of Technology, 1991

Lakshmanan, R. and Geo. Stephanopoulos, "Synthesis of Operating Procedures for Complete Chemical Plants: Part I. Hierarchical, Structured Modeling for Non-Linear Planning," Computers and Chem. Engrg . 12, 985-1002 (1988).

Lakshmanan, R. and Geo. Stephanopoulos, "Synthesis of Operating Procedures for Complete Chemical Plants: Part II. A Non-Linear Planning Methodology," Computers and Chem. Engrg. , 12, p. 1003-1021 (1988).

Lees, F. P., Loss Prevention in the Process Industries, Butterworths, London, 1980

Lees, F.P., "Hazard Warning Structure: Some Implications and Applications", IChemE Symposium Series No. 80, 1983, p. J1

Lowe, D.R., and C.H. Solomon, "Hazard Identification Procedures", IChemE Symposium Series, No. 80, 1983, p. G8

Maher, M. L., "Expert Systems for Structural Design," Expert Systems in Engineering, D. T. Phan, Ed. IFS Publications/Springer-Verlag (1988)

Mosleh, A., Bier, V. M., and G. Apostolakis, "A Critique of Current Practice for the Use of Expert Opinions in Probablistic Risk Assessment," Reliability Engineering and System Safety, 1988, 20, p. 63

Ozog, H. and L.M. Bendixen, "Hazard Identification and Quantification", Chemical Engineering Progress, April, 1987, p. 55

Ozog, H., "Hazard Identification Analysis and Control", Chemical Engineering, February 18, 1987, p. 161

Perkins, J. D. and Barton G. W., "Modelling and Simulation in Process Operation", in *Foundations of Computer-Aided Process Operations*, G. V. Reklaitis and H. D. Spriggs (Editors), CACHE-ELSEVIER (1987)

R. J. Brachman and H. J. Levesque, Readings in Knowledge Representation, MOrgan Kaufmann Pubishers, Los Altos, 1985

Sheil, Beau, "Power Tools for Programmers," *Datamation*, February 1983, pp. 131-144.

Sheil, Beau, "The Artificial Intelligence Tool Box" in Artificial Intelligence Applications in Business, Reitman, W. ed. pp. 113-126., 1984

Slater, Corran, and Pitblado, eds., "Major Industrial Hazards Project Report", The Warren Centre for Advanced Engineering, University of Sidney, May, 1987

Sriram, D. and M. L. Maher, "The Represetation and Use of Constraints in Structural Design," Applications of Artificial Intelligence in Engineering Problems, Volume 1, 1st International Conference Southampton University, U. K., Sriram and R. Adey Eds., April (1986)

Stephanopoulos, Johnston, and Lakshmanan, "An Intelligent System for Planning Plantwide Process Control Strategies," Journal A, 29(3), p. 81-86 (1988).

Stephanopoulos, Johnston, Kriticos, Lakshmanan, Mavrovouniotis and Siletti, "DESIGN-KIT: An Object-Oriented Environment for Process Engineering," Computers and Chem. Engng., 11, p. 655-674 (1987).

Stephanopoulos, George, "The Future of Expert Systems in Chemical Engineering," *Chemical Engineering Progress*, September 1987, pp. 44-51.

Stoessel, F., "Experimental Study of Thermal Hazards During the Hydrogenation of Aromatic Nitro Compounds," 6th International Symposium "Loss Prevention and Safety Promotion in the Process Industries" European Federation of Chemical Engineering, p. 77-1 - 77-14

Walling, C., Free Radicals in Solution, J. Wiley & Sons, 1957, p. 28-29

3.7. Appendix A

3.7.1 Preliminaries

To show that all $2^{|S|}$ subsets of sources can be formed in a plant we will use a graph theoretic argument combined with reasonable assumptions about the nature of processing plants.

A process plant is defined by a graph {V, E}:

- V ≡ Set of process nodes, $\{v_1, v_2 \ldots v_i \ldots v_n\}$.
- $S \subseteq V$ Set of sources.
- $O \subseteq V$ Set of sinks.
- E ≡ Set of connections, $\{e_{ij} \mid (v_i, v_j)\}$.

Process nodes are defined to be places where significant hold up can occur and where changes in thermodynamic state can take place. In addition, there are two special types of nodes, sources and sinks. Sources, S, are the entry points of material into the plant, and correspondingly O are the outlet points. We need to introduce two different types of sources:

1. *Normal Source* This type of source enters at a source nod but can be turned on or off. This is used to model a feed to a plant that can be interrupted, for example a feed that enters via a pipe with a valve.

2. *Unblockable Source* This type of source enters via a route with no means to prevent the flow. Such a source is in reality very unlikely, but as an example, consider an open vessel which will have an unblocked source of air.

Two nodes are connected when there exists a means for the contents of one equipment item to contact those of another.

From these two definitions it is clear that there is no unique graph associated with a process plant. For example a process node could be an individual piece of equipment or the phases that exist within the equipment, and two process nodes could be connected by environmental release. *The validity of the graph theoretic result is independent of the modeling of the plant, its practical relevance is not.*

In addition to associating a graph with the processing plant we have to decide how material accumulates at those nodes, and how the connections between process nodes will allow material to flow between the different nodes. To represent the material we associate a *label* with each node and edge, and for the performance of the connections, a set of *Labelling Rules*. A labelling rule is an assignment operator that assigns tha value to the right of the *rightarrow* to the label to the left. For our purposes we adopt the following set of rules.

$$L(v_i) \rightarrow L(e_{ij}) \rightarrow L(v_j) \forall v_i, v_j \tag{1}$$

$$\bigcup_i L(e_{ij}) \rightarrow L(v_j) \quad \forall v \tag{2}$$

$$L'(v_i) \cup L(v_i) \rightarrow L(v_i) \tag{3}$$

$$\{\} \rightarrow L(v_i) \quad v_i \in O \tag{4}$$

The *rightarrow* symbol denotes assignment. Intuitively 1 establishes the effect of flow in the plant from one piece of equipment to another, stating that if we move material from one node to another the connection must be labelled with this material. The rule can be applied to any pair of nodes and thus implies that there is no directionality to the connections, i.e. the edges of the graph are undirected. Furthermore this rule is local,

flow can be intiated between two nodes without its effect apreading to other nodes. 2 indicates the effect of allowing multiple flows into a single process node: the species set present is simply the union of all the edge labels. 3 establishes the effect of a source, allowing us to augment the set of species already present at a node with the species of the source. For unblockable sources this rule must always be applies, for normal sources we may choose not to invoke it to label the node. 4 describes the effect of a sink at a node, we can essentially 'empty' the node of species. In practive this is impossible, without vacuum equipment, thus the empty state is taken to represent an inert environment.

3.7.2. Proof

With these definitions in place we can now proceed to the main result: that any arbitrary subset of source species can be formed at some point in the plant. The result essentially reduces to requiring that every pair of sources be connected.

$$\forall v_i, v_j v_i \in S v_j \in S \exists p \forall v_k on(p, v_k) \wedge$$

$$(\neg unblockable(v_k) \vee v_i = v_k \vee v_j = v_k)$$

$$\wedge connects(p, v_i, v_j) \Leftrightarrow$$

$$\forall \Omega \exists v_l sourcesubset(\Omega) \wedge at(v_l, \Omega)$$

Proofby contradiction:

$\Leftarrow$

1. Assume lhs

$$\forall\ \Omega\ \exists\ v_l\ sourcesubset(\Omega) \wedge at(v_l, \Omega)$$

2. Assume $\neg rhs$

$$\exists\, v_i, v_j v_i \in S v_j \in S \forall p \exists v_k on(p, v_k) \wedge$$

$$(unblockable(v_k) \wedge v_i \neq v_k \wedge\ v_j \neq\ v_k$$

$$\wedge\ connects(p, v_i, v_j)$$

Consider the subset Ω consisting of the components in source v_i and v_j which satisfy 2.

Every path between the two vertices contains an unblockable source, thus by our labelling rules this source must be included in at any vertex where v_i and v_j contact, thus we cannot construct the subset (v_i, v_j) at any vertex, which contradicts

1. $\Rightarrow$

    1. assume rhs

    2. assume $\neg lhs$

$$\exists\, \Omega\ \forall\ v_l \neg sourcesubset(\Omega) \vee \neg at(v_l, \Omega)$$

Consider all sourcesubsets, thus from 2. above at least one cannot be formed at a node, call it Ω'. Consider the elements of Ω' $(v_i, v_j, \ldots v_n)$, each of these is pairwise connected by 1., thus by adjoining all pairwise paths I can find a vertex where all the elements can be assembled using the labelling rules. Furthermore for each pair there is a path that avoids any unblockable sources that are not part of Ω', and thus no additional elements have to be included, this contradicts 2.

This establishes the necessary and sufficient condiitons for all $2^{|S|}$ subsets to be formed: we must have paths between every pair of source nodes which do not contain unblockable source nodes unless they are of the sources being connected. This implies that the process network must be connected, otherwise sources from one disjoint component are partitioned from any others.

Chapter 4
A Methodology for Automating Hazard Identification and Analysis
Part II: Deductive Determination of the Causes of Hazards

Summary

An intensive process-based methodology has been developed to solve the problem of identifying inherent design weaknesses. The methodology provides a means for quantifying the level of inherent safety associated with design alternatives. The approach begins with the inductive determination of hazardous chemical or physical reactions, or both, and proceeds deductively through the network of processing steps to identify all the root causes initiating a chain of events leading to hazards.

Deductive identification is accomplished by constructing the topological map that satisfies the top level event in the context of the process network. Root causes and their preconditions are then identified and associated with the top level event (i.e. the potential hazard) by tracing the influence paths of the variables that define the state of the top level event. The deductive process beginning at the potential reaction set comes with a guarantee: the complete set of chains of events leading to hazards will be covered by the deductions. Hence, the methodology is complete within the domain of knowledge that identified feasible reactions and the modeling scope which establishes the influence of various variable quantities.

An algorithm is proposed for converting topological maps, which establish the chains of events leading to top level events, into fault trees. The methodology, and select components which comprise it, are illustrated through several case studies.

4.1. Introduction

Hazard identification has always been an integral part of design and operational practice. However, it is still to a large degree an informal process dependent on the experience of those directly involved. The basic objective of all risk assessment techniques is to examine an actual or proposed installation to identify and assess potentially hazardous situations, their possible consequences and associated risks, in order to provide a rational basis for determining where risk reduction measures are needed. Once determined, the selection and configuration of process technologies can be modified to eliminate, reduce, or confine potential pathways of accidental events so as to limit potential losses. Therefore, the intent of any analysis is to provide the design engineer with basic information about safety aspects of the process as well as indicate the reliability of controlling or containing the process. Successful design technology relies on the ability to identify and eliminate inherent design weaknesses before they are incorporated into the final design.

Although many approaches employing various strategies have been presented to systematize hazard analysis in the past few decades, procedural robustness of these methodologies is often constrained by the quality of the information available and the expertise of the individuals involved. Hazard analysis can be no better than the extent to which hazards are recognized in the first place. As a consequence, the engineer's ability to design safety into a process is limited by his understanding of the top level events and the completeness in which those events have been specified as well as his understanding of the pathways which enable those events.

Lees (1980) stated: "The safety of the plant is determined primarily by the quality of the basic design concept rather than by the addition of special safety features". It is difficult to over emphasize this point. The degree to which it is economic to eliminate (as opposed to control) a hazard is very much dependent upon when the hazard is first

recognized. By the time the design has reached the stage of being sufficiently well documented to allow a detailed hazard identification study to be done, the flexibility to eliminate hazards entirely is very much reduced. If the time of hazard/error detection is to be shifted to pre-design review, a less encumbered approach must be developed and integrated into the design process.

4.1.1 Limitations of Past Approaches

The fundamental premise in past approaches which attempt to mitigate or control hazards lies in the assumption that they have the ability to both identify accurately and pinpoint precisely the location of a future hazard. Virtually all experience suggests that the contrary is true. In fact, where we suspect the possibility of a hazard, we take added precautions so that the hazard will not occur. Moreover, we have shown that while understanding the set of enabling conditions is essential for safe plant operation, the identification of the entire set of enabling conditions is an intractable task: it is impossible to completely identify the set of enabling conditions leading to a hazardous state.

This occurs because enabling conditions are a set of variable values, which, when substituted into the equations representing the plant, cause the variables defining the TLE to achieve values necessary for hazard occurrence. Since these variable values are drawn from the set of real numbers we cannot search over all values to determine the variable values explicitly. Shifting to a representation where we bracket variable values to create ranges over which the behavior is safe or unsafe is only a partial solution. Firstly, the ends of the ranges for variables are essentially functions of the values that a given variable achieves. Secondly, if the equations display chaotic solution behavior then we cannot be certain that the system behaves the same way for all numbers within a given finite range.

As a result, we have focused on the interpretation of the pathway leading to a hazardous state and its topography and how these relate to the inherent safety of the design technology rather than on the elucidation of pathways leading to top level events. In the absence of complete enabling condition identification, such an approach is essential if the safety of a chemical operation is to be enhanced.

Consequently, we have been successful in overcoming several limitations of past approaches:

1. establishment of a systematic and formal methodology for increasing the inherent safety of a design technology.

2. providing a means for quantitatively assessing the inherent safety of design alternatives.

3. establishment of a systematic and formal strategy for optimizing the inherent safety of a design technology through the selection of appropriate control points.

4. guaranteeing the correctness and completeness of pathways leading to top level events.

4.1.2 Overview of the Proposed System

In Chapter 3 we presented a methodology for completely identifying the set of hazardous states preceding chemically based top level events. In this chapter we present the methodology for determining the pathways leading to these states.

The methodology is conceptually distinct from traditional approaches in the elucidation of underlying pathways: it begins with the variable set that defines the hazardous state and propagates the influence of these variables to equipment failures (e.g. forward

chaining or goal directed reasoning); rather than beginning with process equipment failures and analyzing their ability to create a potentially hazardous state (e.g. back chaining or hypothesis directed reasoning). Using the variable set defining the hazardous state (i.e. the *(n+2)* independent variables), variable influence pathways that enable the hazardous state are traced using the process equations that specify the design technology.

This strategy is more efficient in the identification of potential pathways leading to the top level event and ensures completeness within the domain of knowledge that identifies the process equations characterizing the design. Furthermore, the deductive process beginning with the variables that define the hazardous state comes with a guarantee: the set of pathways leading to these states will be covered by the deductions.

The combined approach unifies the analysis of potential hazards and provides a formal rationale for determining how and where the integration of safety into a design technology should be accomplished. In addition, the methodology proposed in this chapter has been designed to achieve the following major objectives:

1. establish a formal strategy for the integration of safety into a design technology.

2. establish a means for discriminating among design alternatives with respect to disturbance mitigation.

3. develop a basis for optimizing a design technology with respect to disturbance mitigation.

4. establish a method for automatically constructing the network of pathways leading to a hazardous state.

5. develop a method for creating fault trees from the information associated with the hazardous state and the pathway leading to it.

The methodology extensively employs domain specific modeling languages to identify top level events and the pathways leading to them. The utility, development, and usage of these languages were discussed in Chapter 3.

4.1.3 Outline of Chapter

This chapter presents the deductive component of the methodology described in Chapter 3. It has been organized as follows: Section 4.2 describes the formal methodology and discusses several of its properties; Section 4.3 illustrates the methodology using case studies; Section 4.4 presents a summary of the methodology's characteristics.

4.2. Proposed Methodology

In Section 4.1.2 we presented an overview of the proposed methodology. In this section we will: establish the framework of the methodology; present a method for identifying root causes leading to top level events; identify properties of the approach; and develop an approach for constructing qualitative fault trees from topological mappings.

In Chapter 3 we postulated: "Hazards can only be created by the interaction of the system through its boundaries or the altering of internal restraints of the system, such that the system proceeds towards a new equilibrium state." As a consequence, hazards can only arise through the interaction of two or more states which are not in equilibrium.

Our goal in deductive identification of hazards focuses on the use of the $(n+2)$ variables, where "n" represents the masses of the particular chemical species initially charged and "2" represents two independent variable properties (e.g. temperature and pressure), which characterize the hazardous state. Using these variables the pathway of events leading to a

hazardous state can be identified. The completeness in which these pathways are identified is established by Postulate 2; it is given below for convenience:

Postulate 2: The degree of completeness of the set of internal restraints and exogenous factors specifying a hazardous system and its transformation determines the degree of completeness in which the pathways leading to a hazardous state can be identified.

In other words, the opportunities for preventing a hazard, or for understanding the pathways which lead to it, are limited by the completeness of the restraints and the external variables that specify the hazardous state. These in turn may be limited by the quality of the system description or the modeling effort. Consequently, if the equations describing the interrelationship between variables are described correctly, *the effect of initiating events on a pathway is limited to the changing of variable values described in the pathway.*

4.2.1 Methodological Framework

We begin the elucidation of pathways leading to a potentially hazardous state by first identifying the *(n+2)* independent variables that describe the potentially hazardous state. Since design technologies are describable by the network of process equations that define the interactions of various variable quantities, we can trace out the influence of each variable which is associated with the state preceding the TLE. A variable trace or its influence on other variable quantities defines the pathway leading to the TLE. This pathway or network of pathways is the only means by which the variables describing the state preceding the TLE can be affected.

The influence pathway leading to the top level event is constructed from the structural matrix representing the network of process equations. Causality of the process equations, which establishes the topography of the variable influence pathway, is brought out

through the identification of sources: input variables. These sources or variables are established by *external* conditions such as design constraints, control set points, and across the gate feeds. As a consequence, input variables are generally extensive variables. An exception occurs when invariant intensive variables are associated with a feed. For example, the oxygen concentration in air may be considered invariant when it is being used as a feedstock (e.g. formaldehyde production from air and methanol). Similarly, feedstocks delivered to the process with concentration specifications can become inputs (e.g. 100% methanol). Manual settings by operators are input variables. These include: manual valve manipulations, setpoints of controllers, structure of control loops, starting or shutting of pumps, etc. *As a result all potential failures must be input variables. This result occurs because potential failures are sources in the set of process equations which drive the causality in a particular direction.*

Once a set of input variables has been identified and design constraints specified, output variables are assigned according to these inputs. This assignment establishes the structural relationship between equations. Output variables are those variables with respect to which we solve each equation (i.e. they are internal variables to the set of equations); there must be as many output variables as equations for the system to be fully specified (i.e. zero degrees of freedom). Furthermore, a variable occurring in only one equation must be an output variable. Similarly, an equation with only one variable must assign this variable as its output. Consequently, *variables that specify a state preceding a top level event are output variables.*

For example, consider the structural matrix shown below:

	x1	x2	x3	x4	x5	x6	x7
1			x		x		x
2		x			x	x	x
3	x	x	x	x		x	
4		x		x		x	

where columns represent variables in the defining process equations and rows represent the equations themselves. In accordance with the definitions given above, construction of the variable influence path begins with:

1. Assignment of Input

   Step 1 - assign constants as inputs

   Step 2 - assign invariant intrinsic properties of feeds as inputs

   Step 3 - assign set points as inputs

2. Assignment of Outputs

   Step 1 - assign variables occurring in only one equation as outputs

   Step 2 - assign single variables in equations as outputs

   Step 3 - eliminate assigned variables and corresponding equations

   Step 4 - assign dependent variables of scientific equations and definitions as outputs

   Step 5 - Repeat

In this structural matrix, there are seven variables and four equations. Therefore, three variables must be assigned as inputs to specify the system. They are: x2, x4, and x5. Rewriting the structural matrix, by moving the input variables to the right hand side and delineating them with a vertical line we have:

	x1	x3	x6	x7	x2	x4	x5
1		x		x			x
2			x	x	x		x
3	x	x	x		x	x	
4			x		x	x	

where the left hand side of the structural matrix represents outputs and the right hand side represents inputs. Using the definitions shown above outputs are assigned: x1 appears only in equation 3 and therefore receives its output from that equation; similarly, equation 4 contains a single variable on the left hand side of the structural matrix,

therefore x6 takes its output from equation 4. Through the elimination of rows and columns associated with each output, x3 is identified as taking its output from equation 1, and x7 taking its output from equation 2. This is shown below:

	x1	x3	x6	x7	x2	x4	x5
1		**X**		x			x
2			x	**X**	x		x
3	**X**	x	x		x	x	
4			**X**		x	x	

The variable influence pathway leading to x1 is shown in Figure 4-1. As a consequence, x1 can only be influenced by variables which appear in the pathway: causality is established by the input variables. This occurs because the output set assignment given to the set of equations in the structural matrix is unique. This is made evident by rearranging the equations comprising the structural matrix:

	x1	x3	x6	x7	x2	x4	x5
3	**X**	x	x		x	x	
1		**X**		x			x
4			**X**		x	x	
2			x	**X**	x		x

Unique output set assignment occurs when the structural matrix is triangular: the matrix shown above is triangular because variable x6 takes its output from equation 4. As a consequence, the variable influence pathway leading to the top level output variable (e.g. x1) is also unique; and, can only be affected by inputs which occur along the path.

Alternatively, the network of equations may not be represented by a triangular structural matrix. In this circumstance, the structural matrix may not have a unique output set assignment (i.e. one and only one output assignment); and there may be alternative pathways leading to the variable which describes the state preceding the top level event. Alternative pathways are created when loops exist in the set of process equations necessitating simultaneous solution. For example, in the structural matrix shown below there are three potential pathways leading to the variable set.

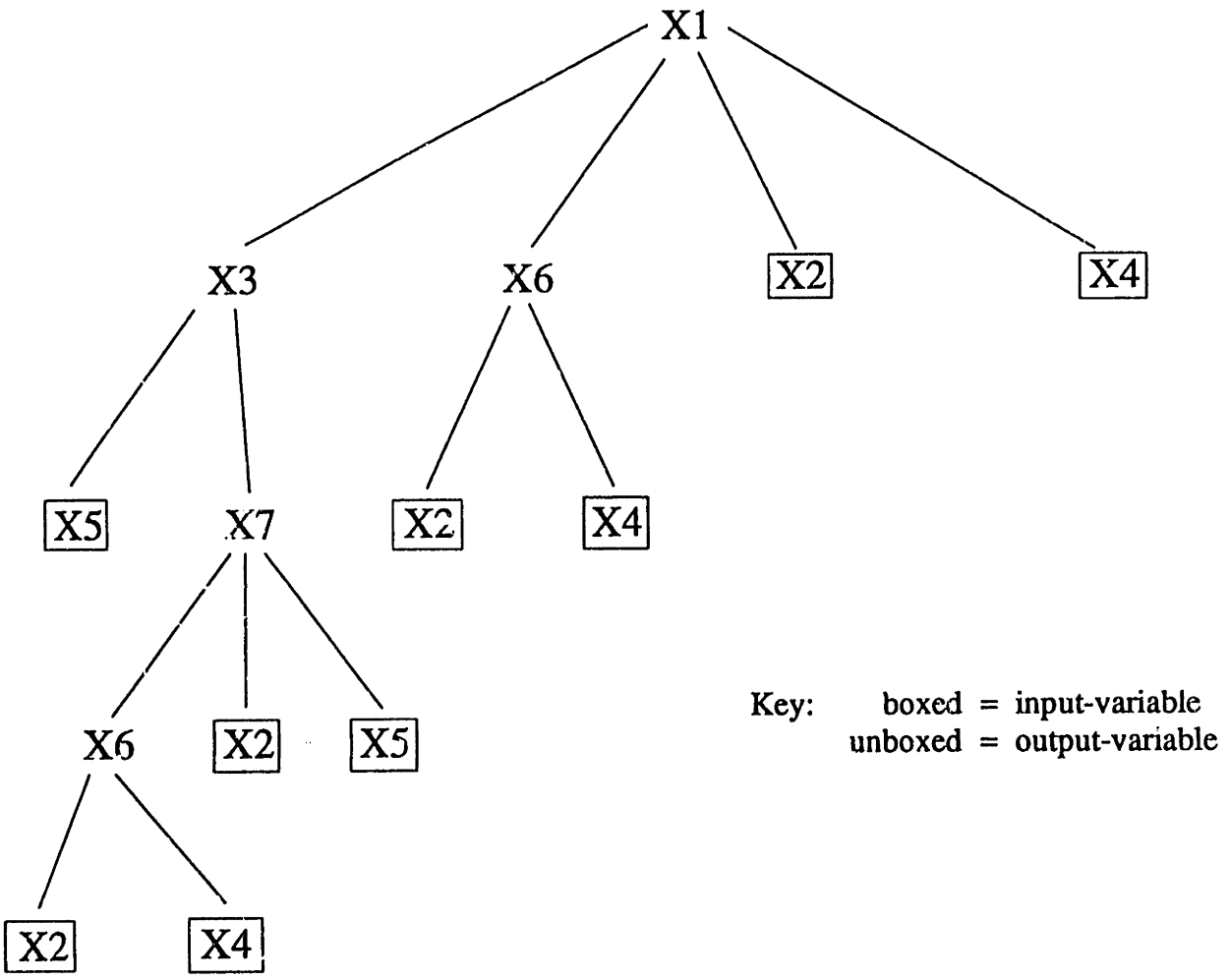


Figure 4-1: Variable Influence Pathway

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1		x	x		x				x			x
2					x		x		x	x		
3				x			x	x	x		x	
4			x	x		x		x	x		x	
5					x	x				x		x
6					x	x	x					x
7								x				
8									x	x		x
9	x	x		x	x		x				x	

Alternative pathways result from the block structure established by variables x5, x6, and x7. Implications of this structure are that the variable x5 can take its output from any one of three equations because no direct cause and effect link exists between x5 and any of these equations. Under these conditions the variable can be tagged and assumptions made about how it obtains its output.

definition: Qualified assumptions come from the boundaries of the specified system: the surroundings of the system; and are clearly related to external variables.

definition: Unqualified assumptions or tentative assumptions rely on contextual interpretations of the process and on the specified system.

Assumptions allow the selection of "external" variables based on engineering judgment, mathematical relationships involving dependent and independent variables (e.g. a rate constant, **k**, is a function of temperature,T: $\mathbf{k} = f(T)$), and abstractions such that they convert a local cyclic system into a linear system. When unique cause and effect assignments cannot be made using assumption based reasoning, each equation which is capable of having multiple outputs must be investigated in turn (see Figure 4-2).

Construction of the pathways or the network routes to the variables describing the potentially hazardous state is achieved using CONSTRUCT-VARIABLE-INFLUENCE-PATHWAYS: a method called from IDENTIFY-POTENTIAL-HAZARD (see Chapter 3). This

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1		X	x		x				x			x
2					X		x		x	x		
3				X			x	x	x		x	
4			X	x		x		x	x		x	
5					x	X				x		x
6					x	x	X					x
7								X				
8									X	x		x
9	X	x		x	x		x				x	

(a)

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1		X	x		x				x			x
2					x		X		x	x		
3				X			x	x	x		x	
4			X	x		x		x	x		x	
5					X	x				x		x
6					x	X	x					x
7								X				
8									X	x		x
9	X	x		x	x		x				x	

(b)

	x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12
1		X	x		x				x			x
2					x		X		x	x		
3				X			x	x	x		x	
4			X	x		x		x	x		x	
5					x	X				x		x
6					X	x	x					x
7								X				
8									X	x		x
9	X	x		x	x		x				x	

(c)

Figure 4-2: Output Assignment Alternatives

method traces the influences of various variables defining the state preceding the top level event. It constructs pathways from the set of process equations contained in the structural matrix so that the relationship of various variables to the top level event can be assessed. The algorithm used for the generation of variable influence pathways is given below:

```
<CONSTRUCT-VARIABLE-INFLUENCE-PATHWAYS>
    input: process-flowsheet
           enabling-criteria
    initialize
      process-flow-equations  ←  apply IDENTIFY-PROCESS-FLOW-
    EQUATIONS
                                 to process-flowsheet
      VIM ← apply IDENTIFY-INFLUENCE-MATRIX to process-flow-equations
      TLE-vars ← apply IDENTIFY-TLE-VARS to enabling criteria
      investigation-list ← TLE-vars
    for each tree in investigation-list
      if next-available-node ← apply IDENTIFY-NEXT-EXPANDABLE-NODE to tree
          then
          expansion-list ← apply EXPAND-NODE to VIM, tree, next-available-
    node
          endif
    return
    for each expansion in expansion-list
      apply CLASSIFY-BRANCH to next-available-node
      if (apply CONTINUE-EXPAND-BRANCH-P to expansion, next-available-node)
          then
          (append tree to investigation-list)
          endif
      return
```

CONSTRUCT-VARIABLE-INFLUENCE-PATHWAYS generates the pathway leading to the top level event in a breadth first manner. EXPAND-NODE is used to establish the expansion list. The algorithm for this expansion is expressed below in pseudocode:

```
<EXPAND-NODE>
    input: VIM, tree, var
    initialize
        expansion-list ← nil
    apply CLEAR-ALL-ASSIGNMENTS to VIM
    for each node in (apply PATHWAY-NODES to tree, var)
        apply ASSIGN-OUTPUT-VARIABLE to VIM, node
        return
    for each equation in (apply EQNS-CONTAINING-VAR to VIM, var)
        if (apply VALID-ASSIGNMENT-P to VIM, eqn, var)
            then
                new-vars ← remove var from (apply VARS-IN-EQN to eqn)
                apply CLASSIFY-VARS to new-vars
                new-tree ← apply APPEND-TREE to tree, node, new-vars
                append new-tree to expansion-list
            endif
        return
    return expansion-list
```

CONSTRUCT-VARIABLE-INFLUENCE-PATHWAYS then applies CLASSIFY-BRANCH to each node in the expansion list. This method classifies each branch according to the technology type which can be used to control the variable value specifying the node. The algorithm for this classification is given below:

```
<CLASSIFY-BRANCH>
    input: node
    for each child-node in (apply CHILDREN to node)
    select case for child-node
        terminal-variable
```

is apply TAG-NO-EXPANSION to child-node
endcase
already-in-pathway
is apply TAG-NO-EXPANSION to child-node
endcase
endselect

select case for node
inherent-controllability
is apply TAG-TYPE-1-TECHNOLOGY to node
endcase
type-2-controllability and process-modification-applied
is apply TAG-TYPE-2-TECHNOLOGY to node
endcase
type-3-controllability and control-loop-applied
is apply TAG-TYPE-3-TECHNOLOGY to node
endcase
endselect
return node

Nodes are classified in this manner to make explicit the protective system responsible for mitigating a disturbance and its relationship to the top level event. CONSTRUCT-VARIABLE-INFLUENCE expands each node of the tree and classifies each branch according to the technology type which could be used to control the variable value specifying the branch in accordance with the following definitions:

definition: A type-1 technology is that technology which is capable of reducing the number of TLEs associated with the design or is capable of modifying the topography of the pathway leading to the TLE through a change in process chemistry or design provided that design changes do not involve the introduction of type-2 or type-3 technologies.

Examples include reactant substitution, catalyst introduction or substitution, byproduct reduction, chemical character modification, solvent substitution, and chemical intermediate elimination. Industrial processes indicative of type-1 technology changes include: reactant substitution in the production of acrylonitrile from hydrogen cyanide and acetylene to propylene, ammonia and air; production of butyl lithium in dilute

solutions prevents spontaneous combustion in the presence of air; direct hydroxylation of ethylene forming ethylene glycol rather than oxidation of ethylene forming ethylene oxide which in turn is hydrated forming ethylene glycol. This eliminates ethylene oxide as an intermediate. Changes in the pathway topography leading to the top level event require design modification, piping changes, equipment substitution, or layout modification.

definition: Type-2 technologies limit the value that a variable involved in the pathway leading to a TLE can achieve without requiring an action.

Examples involving type-2 technologies include: reducing inventories, minimizing operating (process condition) extrema, reducing intermediate accumulation. Industrial processes incorporating type-2 technologies include: utilization of *in situ* reactions to limit intermediate accumulation (e.g. methyl isocyanate consumption via *in situ* reaction reduces the need for storage); utilization of continuous processes to minimize process hold-up (e.g. continuous nitration processes eliminated the need for batch manufacturing of nitroglycerine); reduction in the intensity of processing conditions can lead to smaller material and energy releases (e.g. catalyst improvements has enabled the Oxo process to produce aldehydes from syngas and olefins at lower pressures.

definition: Type-3 technologies limit the value that a variable involved in the pathway leading to a TLE can achieve and require an action to do so.

Type-3 technologies involve active control of process variables in a direct or indirect manner to limit the value that they can achieve. Examples include manipulation of feed rates, cooling water, temperature control, pressure control, catalyst activity assessment by conversion and selectivity monitoring.

The result of technology type tagging is shown in Figure 4-3. Association of different technology types, for the purpose of directly or indirectly controlling the variable values

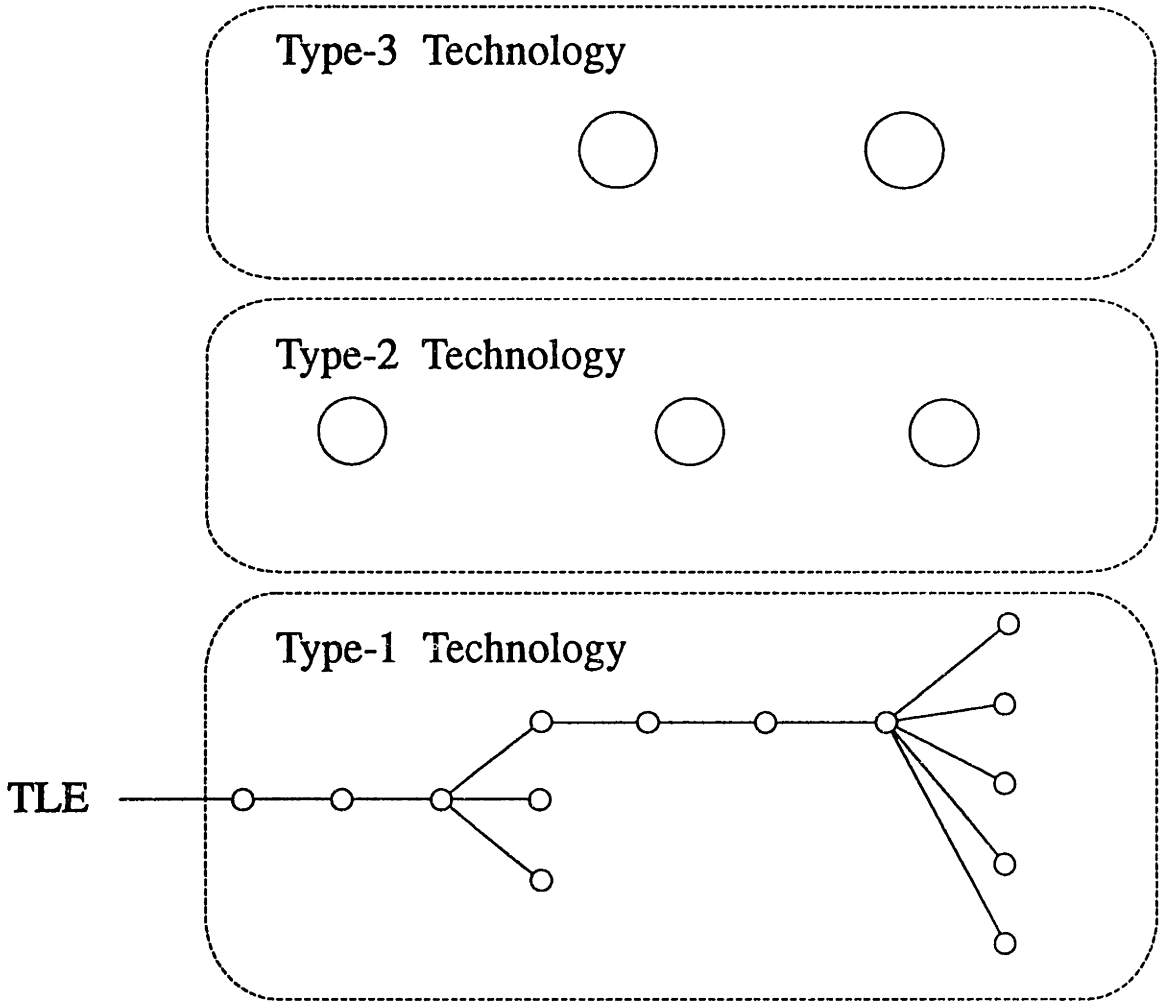


Figure 4-3: Use of Technology Types In TLE Prevention

composing the variable influence pathway to the top level event, is then specified. This specification involves not only an understanding of the controllability of each variable along the pathway but also an understanding of its feasibility with respect to operational criteria.

The strategy used for specifying control objectives is based on closeness: objectives involving variables which are the minimum distance from the top level event are preferred over objectives involving variables which are more distant. However, the point of manipulation (e.g. how this objective is satisfied) remains unspecified: it is dependent on the type of disturbance to be mitigated. This strategy reflects the necessity that foreseen disturbances must enter the variable influence pathway through the input variables. As a consequence, the propagation of effects through input variables at level *n* eventually passes through variables at level 1 to enable the top level event; therefore, it is safer, and preferred, to have control objectives at level 1 before level 2 and so on to level "*n*". It is preferred, therefore, that the manipulated variables which achieve the control objective be located at the inputs (i.e. level *n*) so that it can mitigate the disturbance before it has the opportunity to amplify (see Figure 4-4).

definition: Foreseen disturbances enter the process through the set of external variables or inputs. These include pump failures, valve failures, controller malfunctions, etc.

Notice, the greater the distance between the control objective and the top level event, the greater the opportunity (i.e. the higher the probability) for introducing a disturbance along the pathway connecting the control objective to the TLE; thereby negating the control objective. These disturbances may be foreseen, occurring after the control objective through an input variable of the process, or unforeseen.

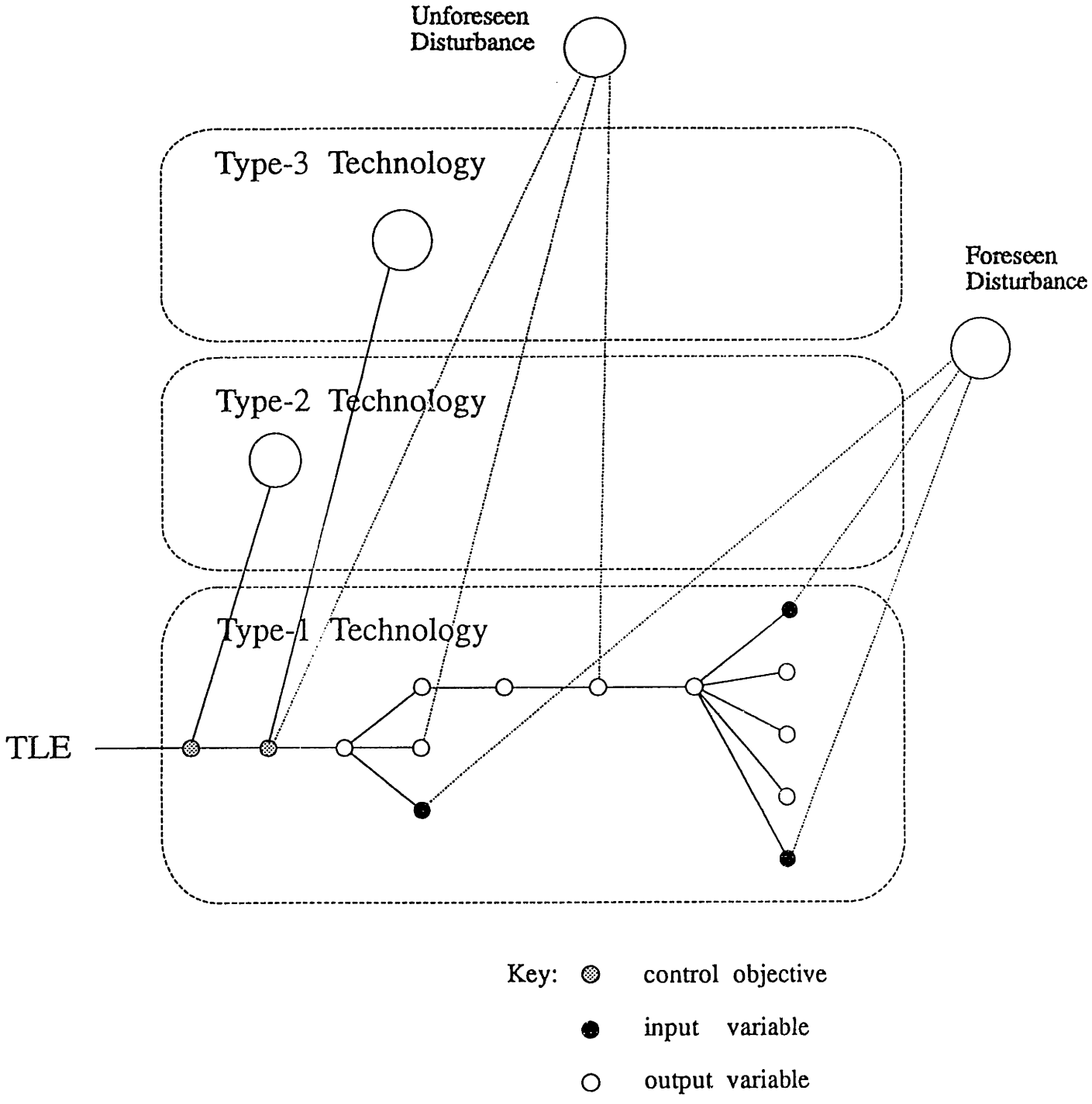


Figure 4-4: Disturbance Mitigation Through Control Point Specification

definition: Unforeseen disturbances can enter the process as a source or input variable anywhere along the pathway and often change the pathway leading to the top level event. They are unpredictable and often manifest themselves as boundary failures.

A result of this definition is that variable influence diagrams are useful in establishing root causes leading to a top level event only when disturbances are foreseeable.

The methodology used for GLOBAL-HAZARD-IDENTIFICATION (see Chapter 3) enables the identification of TLEs independent of the pathways which lead to them. TLEs are identified by GLOBAL-HAZARD-IDENTIFICATION as it expands the scope of the system description in its search for potential hazards. System expansion affords boundary failure simulation thereby anticipating the effects of unforeseeable disturbances. Moreover, since the thermodynamic state preceding a top level event is specified by IDENTIFY-POTENTIAL-HAZARD, any disturbance, foreseeable or unforeseeable, must act as an input to these variables, if the TLE is to be enabled. Consequently, control objectives involving TLE variables can be used to mitigate unforeseeable disturbances.

Type-2 technologies can be particularly effective in mitigating unforeseeable disturbances because they do not require active control of the disturbance. Alternatively, for pragmatic reasons, type-3 technologies are preferred for the control of foreseeable disturbances; while type-1 and type-2 technologies are preferred for the control of unforeseeable disturbances. The latter is accomplished by adding or removing equations from the variable influence pathway or by changing the chemistry of the process. Figure 4-4 illustrates these points.

Integration of technology types for TLE prevention is illustrated in Figure 4-5. The classification of variable controllability by technology type affords a utility for determining whether a design technology's inherent safety is improving by explicitly establishing its ability to mitigate disturbances.

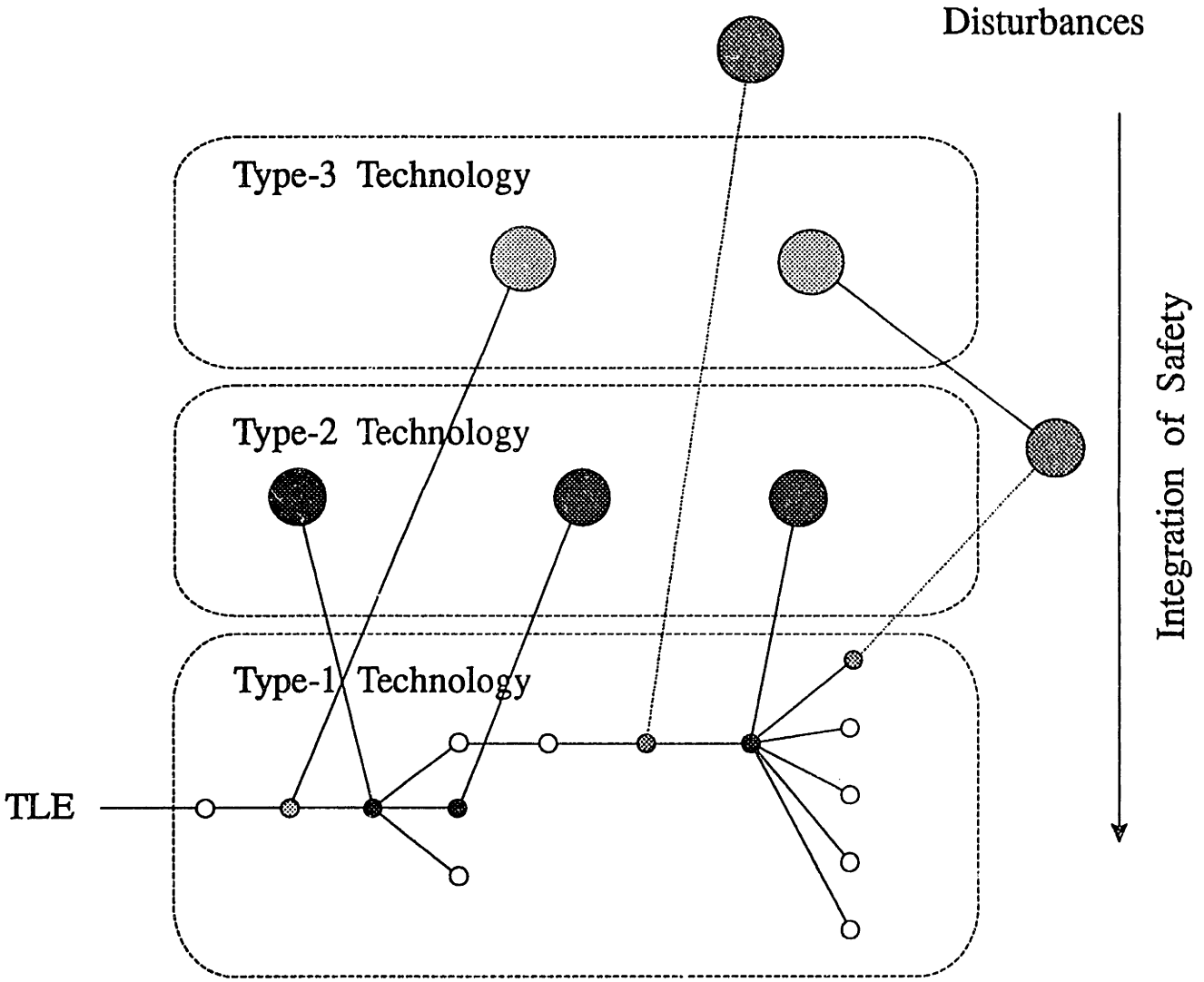


Figure 4-5: Integration of Technology Types for TLE Prevention

Root causes which lead to a top level event are constructed using information contained in this hierarchical representation in conjunction with knowledge obtained from the process node representation (see Chapter 3), and enabling-criteria of the hazard forming reaction. For example, restricting our attention to foreseeable disturbances, the TLE shown in Figure 4-6 may be enabled by the failure to achieve control objective(s). Any combination of failures may lead to the top level event; the effect of various failure combinations requires quantitative analysis.

IDENTIFY-NONDISSIPATIVE-PATHWAYS constructs the set of enabling conditions that lead to a top level event. This method takes as its input classified-influence-pathways which it obtains from CONSTRUCT-VARIABLE-INFLUENCE-PATHWAYS. TLE variables are then identified and each variable contained in the set is traced to identify potentially feasible roots. When an achievable root is identified (i.e. an input disturbance, controlled or uncontrolled, which can enable the top level event) it is collected into root causes and returned. The algorithm used by this method is shown below:

```
<IDENTIFY-NONDISSIPATIVE-PATHWAYS>
    input:      classified-influence-paths
    initialize: TLE-variables ← (collect-TLE-variables
                                    classified-influence-paths)
    for each variable in TLE-variables
        when (unique-path-p classified-influence-paths)
            (apply CONSTRUCT-FEASIBLE-ROOTS to TLE-variables
                classified-influence-paths)
              and collect into root-causes
            return
        else
            for each path in classified-influence-paths
            to (apply CONSTRUCT-FEASIBLE-ROOTS to TLE-variables path)
            and collect into root-causes
            return
        return
    end
```

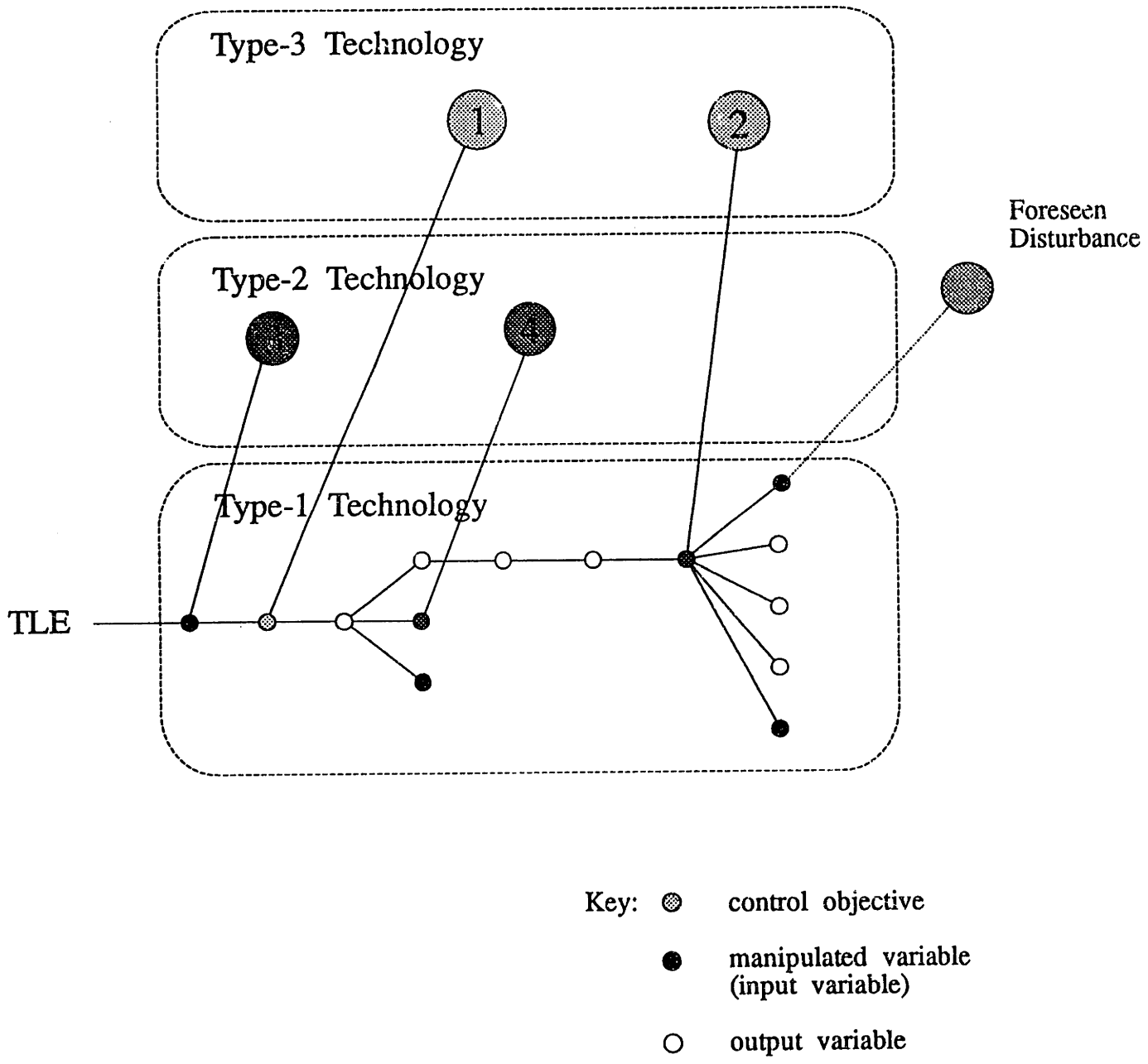


Figure 4-6: Technology Placement for TLE Prevention

The construction of feasible roots: branches which can enable; is generated when IDENTIFY-NONDISSIPATIVE-PATHWAYS calls CONSTRUCT-FEASIBLE-ROOTS. This method assesses the feasibility of a particular variable trace to enable the top level event given a disturbance at its input. CONSTRUCT-FEASIBLE-ROOTS identifies and associates the controller responsible for mitigating the input disturbance. When a means of control is not associated with the input, the variable and its pathway are collected directly into roots. The algorithm used for branch construction is:

```
<CONSTRUCT-FEASIBLE-ROOTS>
    input: variable-set
           pathway
    when (enabling-feasibility-p variable-set pathway)
       for each variable in variable-set
           when (controlled-variable-p variable pathway)
              controller ← (get-controller variable pathway)
              collect variable, pathway, and controller into roots
              return
           else
              collect variable into roots
           return
       return
    return
    end
```

4.2.2 Fault Tree Construction

Using this information and knowledge associated with the potentially hazardous state (i.e. reaction information), a qualitative fault tree can be constructed. CONSTRUCT-QUALITATIVE-FAULT-TREE takes as its input the top level event, the associated hazardous state preceding the TLE, and the root causes for each variable defining the hazardous state. Using these inputs, it determines the reactions responsible for the hazardous state and the enabling criteria of the reaction. With this information it constructs a level-1-gate: a logical gate immediately preceding the top level event. This gate establishes the demands (i.e. inputs) of the top level event. By looping through the set of demands associated with the level-1-gate, qualitative logical gates can be

constructed from a given demand input and its root causes. Since each qualitative gate has associated with it an input and an output, these can be linked together forming a tree. By linking the tree to the level-1-gate and in turn appending the level-1-gate to the TLE a qualitative fault tree can be constructed. The algorithm used is:

```
<CONSTRUCT-QUALITATIVE-FAULT-TREE>
    input: TLE
           hazardous-state
           root-causes
    initialize: hazard-variables ← (get-hazardous-variables hazardous-
    state)
                reaction ← (get-reaction hazardous-state)
                enabling-criteria ← (get-enabling-criteria reaction)
                level-one-gate ← (construct-level-one-gate
                                        hazard-variables enabling-criteria)
    for each input in level-one-gate
       gates ← (CONSTRUCT-QUALITATIVE-GATES input (get-root-causes
    input))
       tree ← (LINK-GATES input gates)
       append tree to level-one-gate
       return
    append TLE to level-one-gate)
    return
    end
```

Basic gates are constructed using CONSTRUCT-QUALITATIVE-GATES. This method builds or-gates, and-gates, and special-gates. Special-gates require quantitative analysis to determine their logical basis. It accomplishes this given an input variable and the root causes associated with that variable. This method uses the variable trace associated with the root causes to establish the pathway leading to the external input and the protective devices associated with that input. By collecting the controllers associated with the various controlled variables, and-gate, or-gates, and special-gates are constructed. The algorithm used for CONSTRUCT-QUALITATIVE-GATES is shown below:

```
<CONSTRUCT-QUALITATIVE-GATES>
    input: variable
           root-causes
```

```
initialize: and-gates ← nil
            or-gates ← nil
            special-gate ← nil
            initial-internal-input ← (get-internal-input variable root-
causes)
for each variable in (get-output-assignment root-causes)
    upfrom initial-internal-input
    when (unbranched-node-p variable)
        controller ← (get-controller variable root-causes)
        when controller
            collect (construct-and-gate variable controller (get-input
variable))
                into and-gates
            and
            collect (construct-or-gate variable controller (get-input
variable))
                into or-gates
            return
    collect (construct-special-gate variable (get-inputs variable))
        into special-gates
        return
    gates ← append and-gates or-gates special-gates
    return
return
end
```

Complex qualitative fault trees are constructed using the gates identified by CONSTRUCT-QUALITATIVE-GATES. Since each logical gate has a set of inputs and an output, they can be linked together by matching gate outputs to gate inputs. A recursive algorithm is used for this process and is shown below:

```
<LINK-GATES>
        input: outputs
               gates
        while outputs
        for each output in outputs
            sort gates by output
                gs ← (get-gates-with-correct-output output gates)
                g-inputs ← (get-gate-demand-inputs gs)
                collect (apply construct-tree output gs)
                into tree
            and
```

```
            apply (link-gate  gs  g-inputs)
        return
    return
    end
```

Qualitative analysis of the pathway leading to the TLE is required to establish the logical basis for converting special-gates into structures (i.e. trees) containing or-gates and and-gates. By associating averaged probability and failure rate data with the resulting qualitative fault tree (i.e. with the variables and the technology types) it can be transformed into a conventional fault tree. Construction of the fault tree in this manner has particular utility because it is complete: all pathways leading to the TLE will be identified within the scope of the equations describing the design technology. This prevents incompleteness in the pathways leading to the TLE and minimizes errors[1] in the frequency of that event; often by more than three orders of magnitude [ICI, 1988].

4.3. Examples in Reaction Based Hazard Identification

In this section, we demonstrate how the methodology can be used to identify root causes through a series of case studies. We focus on the identification of enabling conditions and the construction of fault trees.

4.3.1 Case Study 1: reactor quench

The process shown in Figure 4-7 consists of a reactor which is used for the processing of a highly unstable chemical which is sensitive to small increases in temperature. The reactor is equipped with a quench tank to protect the system against a runaway reaction and is monitored by two temperature sensors: T_1 and T_2. Sensor T_1 automatically activates the quench tank outlet valve when it detects a temperature rise above the specified set point. Sensor T_2 sounds an alarm in the control room to alert the operator to the process upset. When the alarm sounds, the operator closes the reactor inlet valve.

---

[1]ICI has shown that incompleteness can result estimates of the TLE that are off by as much as three to five orders of magnitude; while, errors in probability and failure rate data lead to estimates that are within a factor of two to five.

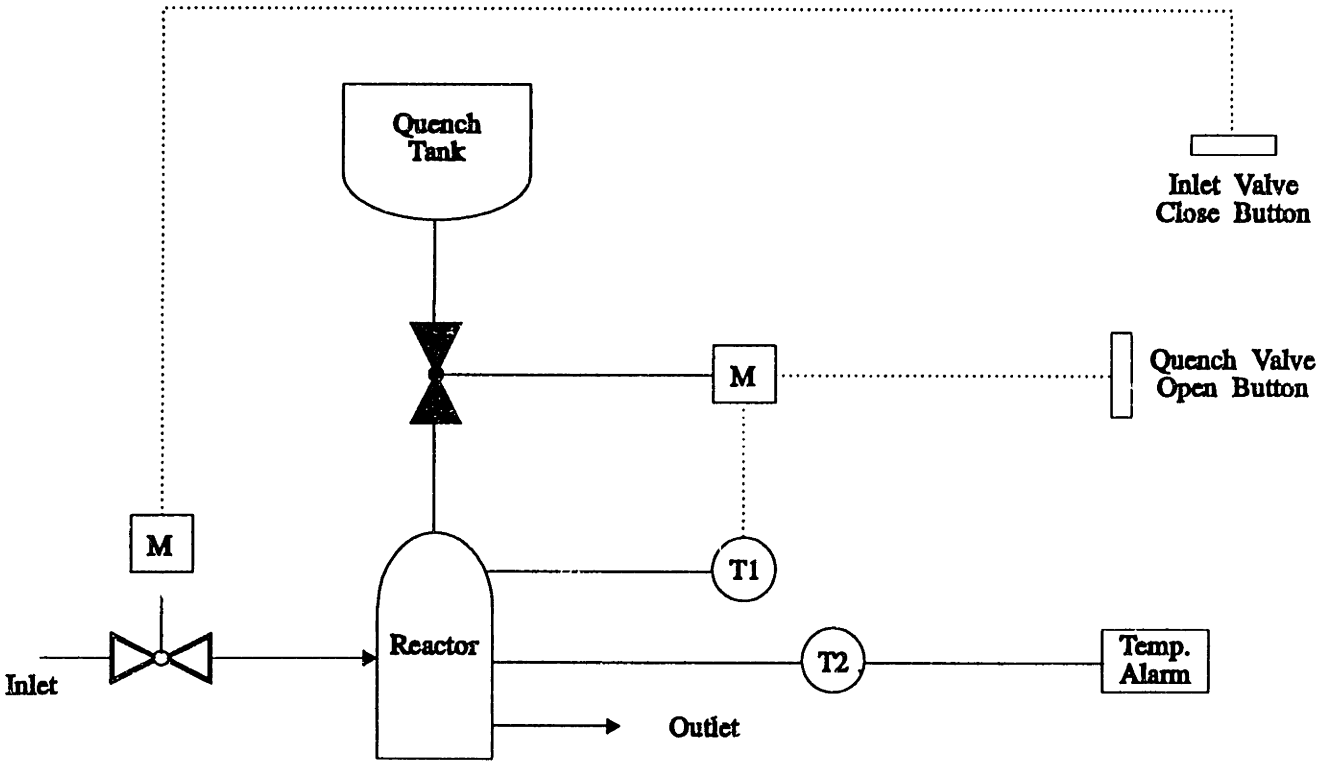


Figure 4-7: Process Flowsheet

The operator also pushes a quench tank valve button in the control room in case the quench valve fails to open.

The analysis for the deductive determination of root causes begins by constructing the set of process equations which describe the flowsheet. These are shown in Figure 4-8; therein: "F" denotes a feed; "f" denotes a molar flow rate; "Q" denotes the quench feed; "r_A" is a rate expression; "V_{RE}" is the volume of the reactor; "SP" denotes valve stem position; "T" denotes temperature; "Op" denotes an operator. Complex relationships involving these variables show only functional dependency (e.g. equations [9]-[17]). The structural matrix constructed from this set of equations is given in Figure 4-9. Each numbered row describes the relationship corresponding between the equation and the variables which are contained in it; these are represented by columns in the structural matrix.

The process of constructing the variable influence pathway, which leads to the state preceding the top level event (see Chapter 3), begins with the identification of input variables. Following the procedures outlined in Section 4.2.1, input assignment is initiated. Figure 4-10 illustrates how an input specification can be arrived at through the use of qualified assumptions. By restricting our attention to the reactor, examination of the process equations identifies two system constants: Δh and V_{RE}; these become input variables in the structural matrix. Using a qualified assumption involving external feeds, the scope of the system shown in Figure 4-10 can be expanded. This expansion identifies the invariant intensive properties describing the inlet feed (T_F) and the quench feed (T_Q) as external or input variables (see Figure 4-11). Step 3 of the input assignment procedure assigns the setpoints T_2 and T_1 as input variables. The context of the qualified assumption which leads to this specification is shown in Figure 4-12. The structural matrix after input set assignment is shown in Figure 4-13.

[1] $F_2 = f_2^A + f_2^B + f_2^C$

[2] $f_2^C = Q + f_1^C$

[3] $f_1^A = f_2^A + r_A$

[4] $f_1^B = f_2^B - r_A$

[5] $F_1 = f_1^A$

[6] $f_1^B = 0$

[7] $f_1^C = 0$

[8] $r_A = k[C_A]V_{RE}$

[9] $k = A*\exp(-E^*/RT_R) = f(T_R)$

[10] $[C_A] = f(f_1^A, Q, r_A)$

[11] $Q_r = f(\Delta h_r, r_A)$

[12] $0 = f(Q, T_Q, F_1, T_F, Q_r, F_2, T_R)$

[13] $Q = f(SP_1)$

[14] $SP_1 = f(T_R, T_1)$

[15] $F_1 = f(SP_2)$

[16] $SP_2 = f(Op_2)$

[17] $Op_2 = f(T_R, T_2)$

Figure 4-8: Process Equations

	F_1	F_2	Q	f_1A	f_1B	f_1C	f_2A	f_2B	f_2C	r_A	k	C_A	T_R	Q_R	V_{RE}	T_2	T_1	O_{p2}	SP_1	SP_2	Δh_r	T_Q	T_F
1		x					x	x	x														
2			x			x			x														
3				x			x			x													
4					x			x		x													
5	x			x																			
6					x																		
7						x																	
8										x	x	x			x								
9											x		x										
10			x	x						x		x											
11										x				x							x		
12	x	x	x										x	x								x	x
13			x																x				
14													x				x		x				
15	x																			x			
16																		x		x			
17													x			x		x					

Figure 4-9: Structural Matrix

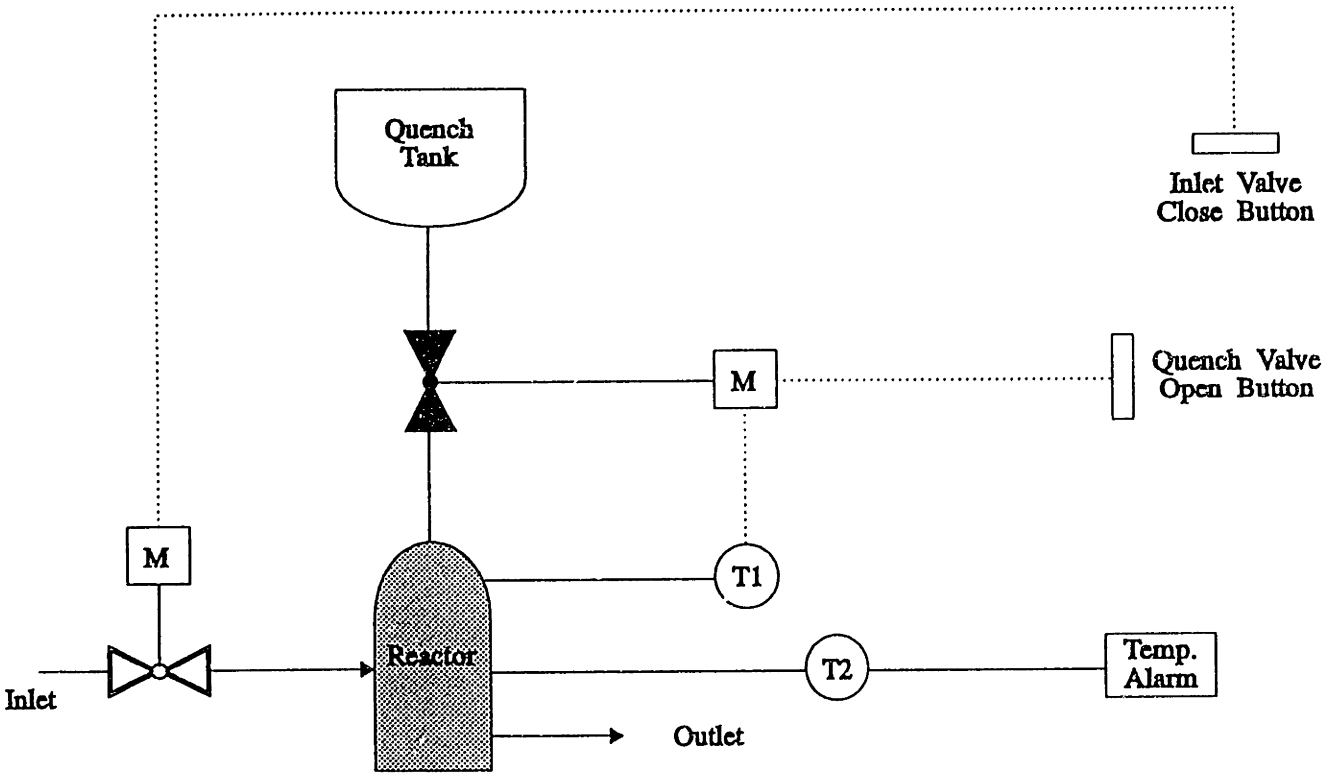


Figure 4-10: Input Specification: Qualified Assumption 1

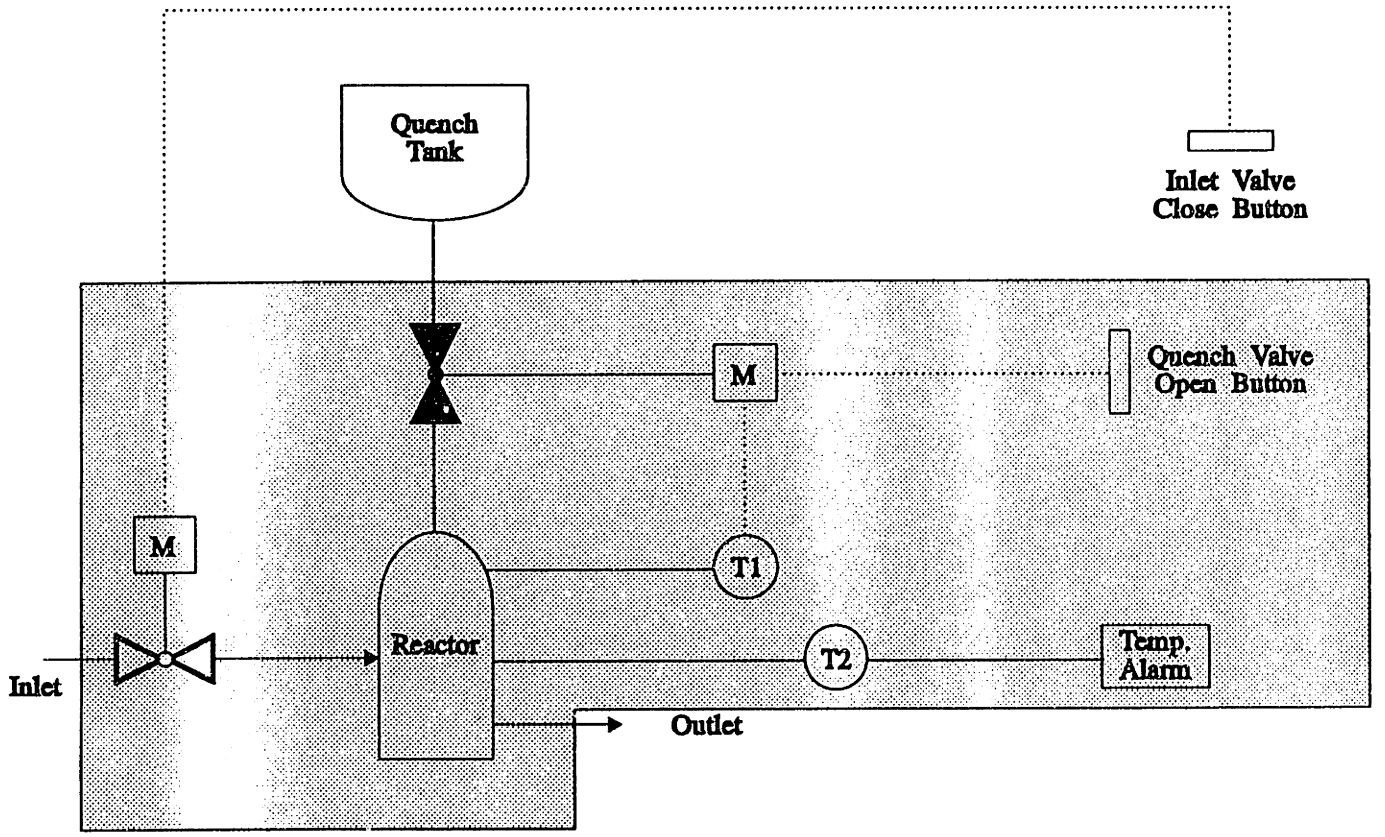


Figure 4-11: Input Specification: Qualified Assumption 2

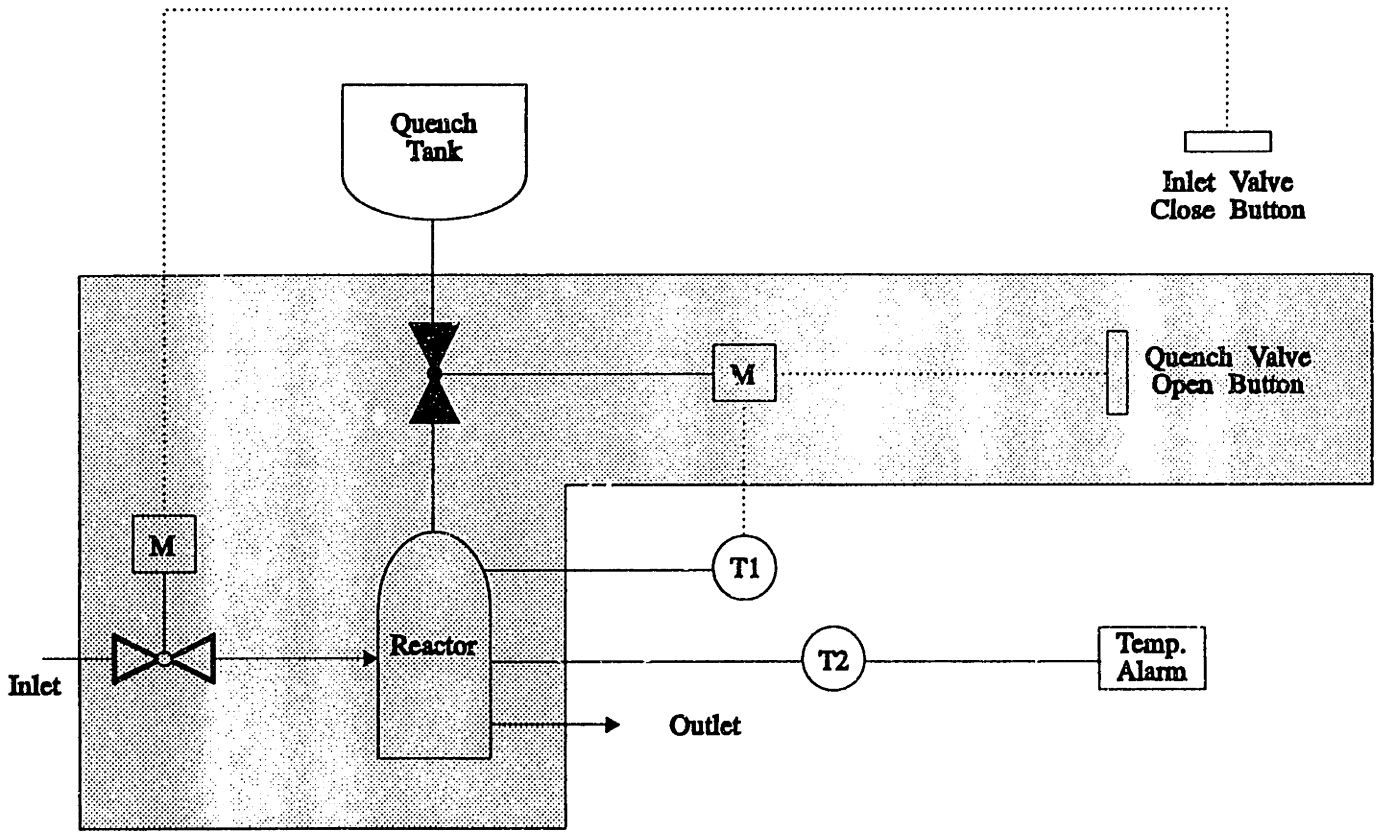


Figure 4-12: Input Specification: Qualified Assumption 3

	F_1	F_2	Q	f_1A	f_1B	f_1C	f_2A	f_2B	f_2C	r_A	k	C_A	T_R	Q_R	O_{p2}	SP_1	SP_2	Δh_r	T_Q	T_F	V_{RE}	T_2	T_1
1		x					x	x	x														
2			x			x			x														
3				x			x			x													
4					x			x		x													
5	x			x																			
6					x																		
7						x																	
8										x	x	x									x		
9											x		x										
10			x	x						x		x											
11										x				x				x					
12	x	x	x										x	x					x	x			
13			x													x							
14													x			x							x
15	x																x						
16															x		x						
17													x		x							x	

Figure 4-13: Structural Matrix: input assignment

Once inputs are assigned, output assignment begins using the method described in Section 4.2.1. Using this procedure, f_1^B is identified as taking its output from equation [6] and f_1^C is identified as taking its output from equation [7] (see Figure 8). The respective rows and columns of the structural matrix are then eliminated (see Figure 4-14). By using the definition **k**, the rate constant, a qualified assumption can be made that assigns **k** as an output of equation [9] (see Figure 4-15). Upon elimination of this row and column, no direct cause and effect links are identified. The block structure which results allows the remaining variables to take their output from any one of several equations.

To break this loop we make a tentative assumption that Op_2, the variable denoting the operator, obtains its output from equation [17]. This implies that the operator responds to the temperature in the reactor (T_R) and not vice versa. This assumption is then catalogued, as Assumption-1, so that it may be retracted at a later stage as required. Such retraction allows the causality of the set of process equations to be modified. By repeating the procedure specified for output assignment, SP_2 is identified as obtaining its output from equation [16]; and F_1 obtains its output from equation [15] (see Figure 4-16). The set of assignments shown in Step 3 (see Figure 4-16) establish the causality between: reaction temperature and operator; operator and feed valve stem position; and, feed valve stem position and feed rate. Elimination of these respective rows and columns identifies further that f_1^A takes its value from equation [5].

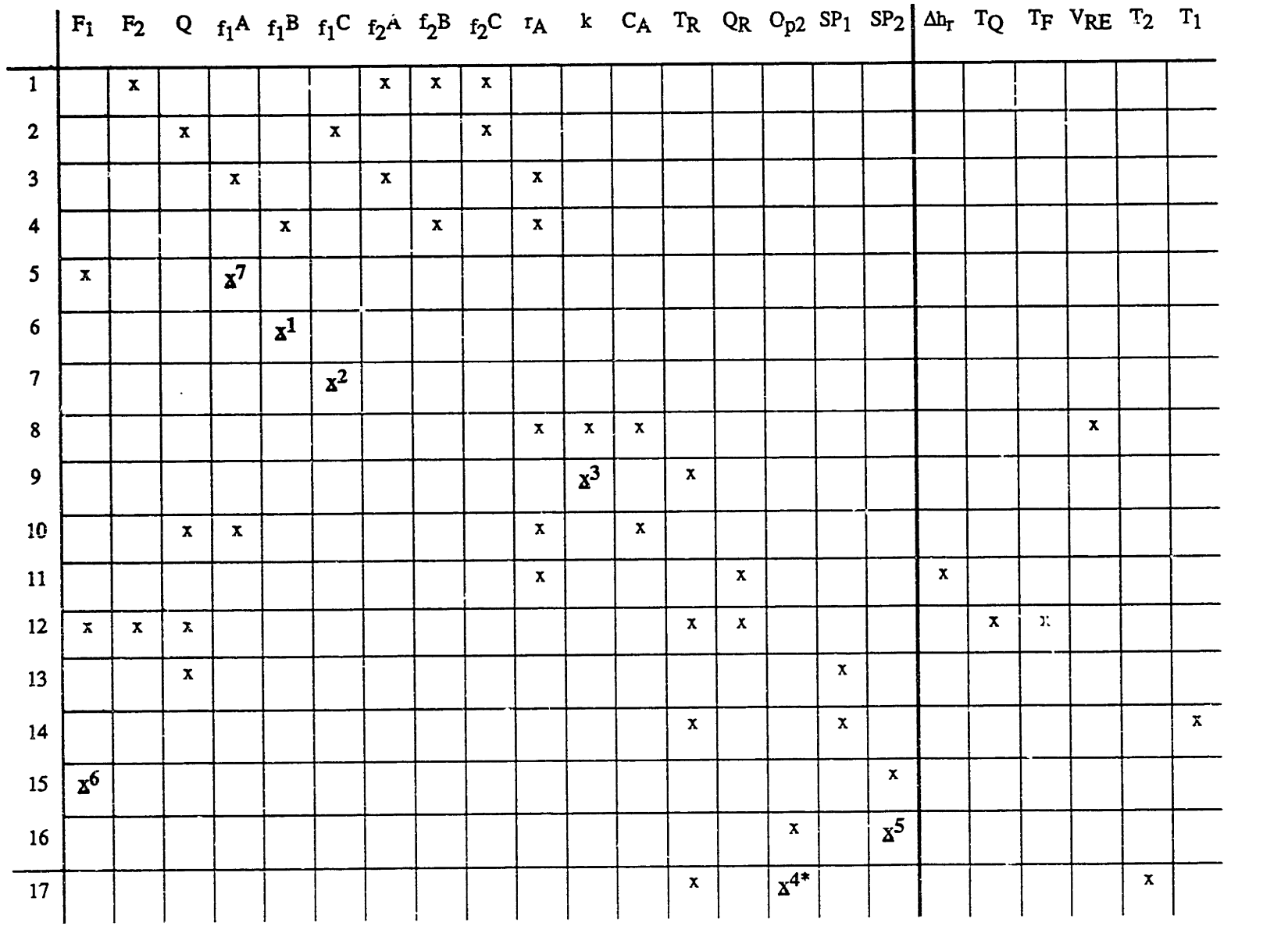
After this assignment a block structure still remains. Again though, a tentative assumption can be made regarding the causality in equation [14]. The assumption is that SP_1, the variable describing the feed valve stem position of is driven by reactor temperature. This assumption is catalogued and denoted Assumption-2. By repeating the process, the output assignment shown in Figure 4-17 results. Notice that variables r_A and C_A have not been assigned. Although an assumption could have been made, specify

	F_1	F_2	Q	f_1A	f_1B	f_1C	f_2A	f_2B	f_2C	r_A	k	C_A	T_R	Q_R	O_{p2}	SP_1	SP_2	Δh_r	T_Q	T_F	V_{RE}	T_2	T_1
1		x					x	x	x														
2			x			x			x														
3				x			x			x													
4					x			x		x													
5	x			x																			
6					x^1																		
7						x^2																	
8										x	x	x									x		
9											x		x										
10			x	x						x		x											
11										x				x				x					
12	x	x	x										x	x					x	x			
13			x													x							
14													x			x							x
15	x																x						
16															x		x						
17													x		x							x	

Figure 4-14: Structural Matrix: output assignment (Step 1)

	F_1	F_2	Q	f_1A	f_1B	f_1C	f_2A	f_2B	f_2C	r_A	k	C_A	T_R	Q_R	O_{p2}	SP_1	SP_2	Δh_r	T_Q	T_F	V_{RE}	T_2	T_1
1		x					x	x	x														
2			x			x			x														
3				x			x			x													
4					x			x		x													
5	x			x																			
6					x[1]																		
7						x[2]																	
8										x	x	x									x		
9											x[3]		x										
10			x	x						x		x											
11										x				x				x					
12	x	x	x										x	x					x	x			
13			x													x							
14													x			x							x
15	x																x						
16															x		x						
17													x		x							x	

Figure 4-15: Structural Matrix: output assignment (Step 2)



Key: * ≡ assumption-1

Figure 4-16: Structural Matrix: output assignment (Step 3)

	F_1	F_2	Q	f_1A	f_1B	f_1C	f_2A	f_2B	f_2C	r_A	k	C_A	T_R	Q_R	O_{p2}	SP_1	SP_2	Δh_r	T_Q	T_F	V_{RE}	T_2	T_1
1		x[12]					x	x	x														
2			x			x			x[10]														
3				x			x[13]			x													
4					x			x[14]		x													
5	x			x[7]																			
6					x[1]																		
7						x[2]																	
8										x	x	x									x		
9											x[3]		x										
10			x	x						x		x											
11										x				x[15]				x					
12	x	x	x										x[11]	x					x	x			
13			x[9]													x							
14													x			x[8≠]							x
15	x[6]																x						
16															x		x[5]						
17													x		x[4*]							x	

Key: * ≡ assumption-1

Key: ≠ ≡ assumption-2

Figure 4-17: Structural Matrix: output assignment (Step 8)

the assignment of these variables in the respective equations, the assumption would have been weak and is not necessary for our purposes.

The variable influence diagram resulting from this assumption set (i.e. Assumption-1 and Assumption-2) is shown in Figure 4-18. This diagram makes explicit pathways leading to T_R and illustrates how disturbances can propagate through the network of equations to enable the TLE. Figure 4-19 illustrates the variable influence diagram for Assumption Set 2: a result of retracting Assumption-1. By collecting the entire set of graphs that describe the set of cause and effect relationships, complete identification of pathways leading to the TLE can be guaranteed, within the scope of the modeling effort.

Through the association of technology types with variables contained in the variable influence diagram the pathway of root causes is constructed. Figure 4-20 illustrates this association for Type-1 technologies to the variable influence diagram described by Assumption Set 1. Notice, that the heat of reaction, Δh, is the only Type-1 technology associated with the TLE. This implies that a potential pathway for mitigating the TLE is to change the heat of reaction; and obviously implies a change in chemistry. Furthermore, the variable influence diagram makes explicit that top level event prevention necessitates the placement of control objectives on every variable that immediately precedes the top level event: T_Q, T_F, Q_R, Q, F_2, F_1.

Association of Type-2 technologies with variables described in the pathway leading to the top level event is shown in Figure 4-21. The value of the variables that are described by Type-2 technologies are generally design specifications and affect the magnitude of the top level event. For example, V_R specifies the volume of the reactor. Similarly, feed and quench temperature vary the rate at which a TLE is achieved. Likewise, so may f_1^B and f_1^C: values describing the molar flow rates of the product and the quench, respectively, contained in the feed stream. This is particularly so if the reaction

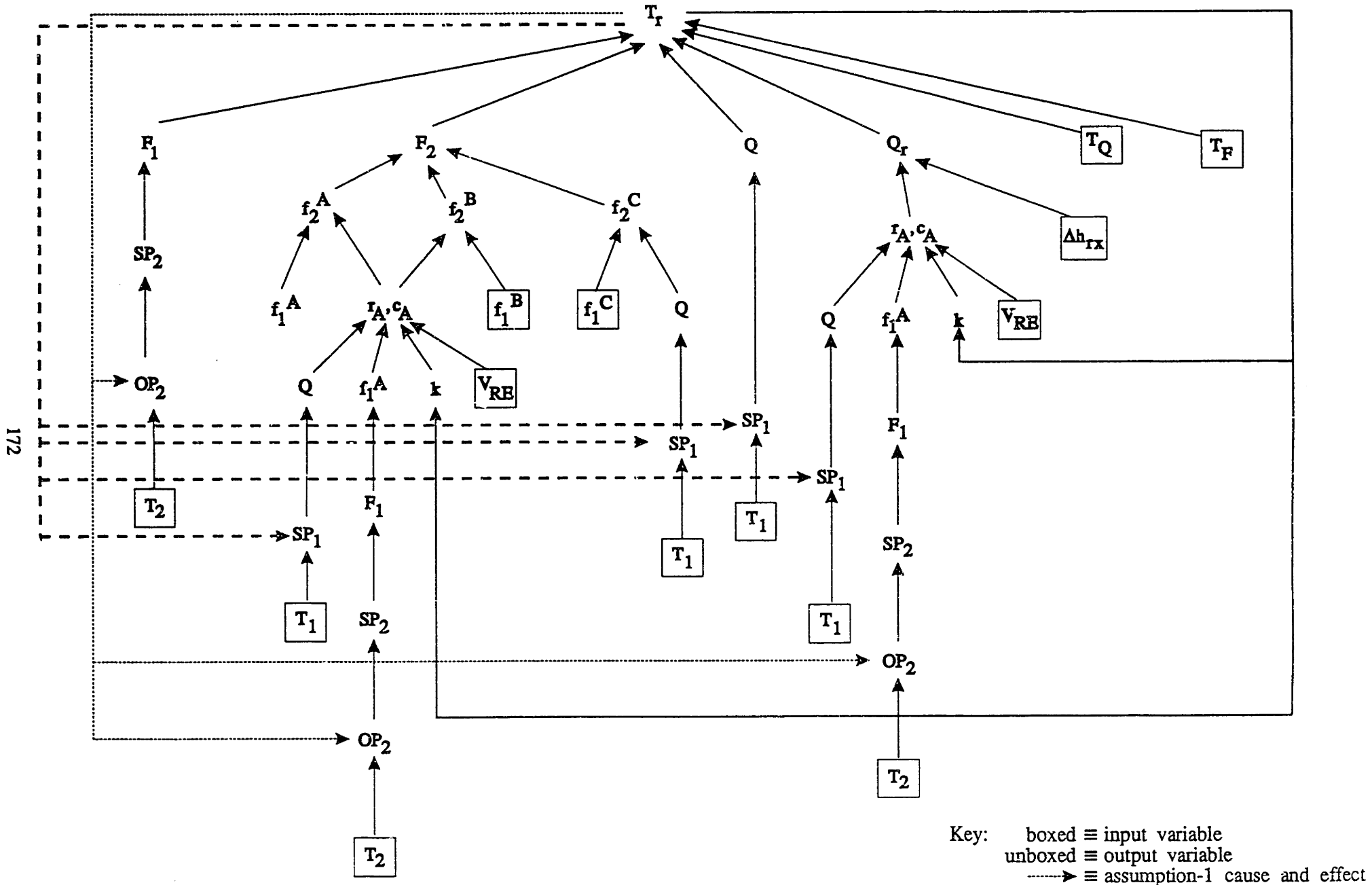
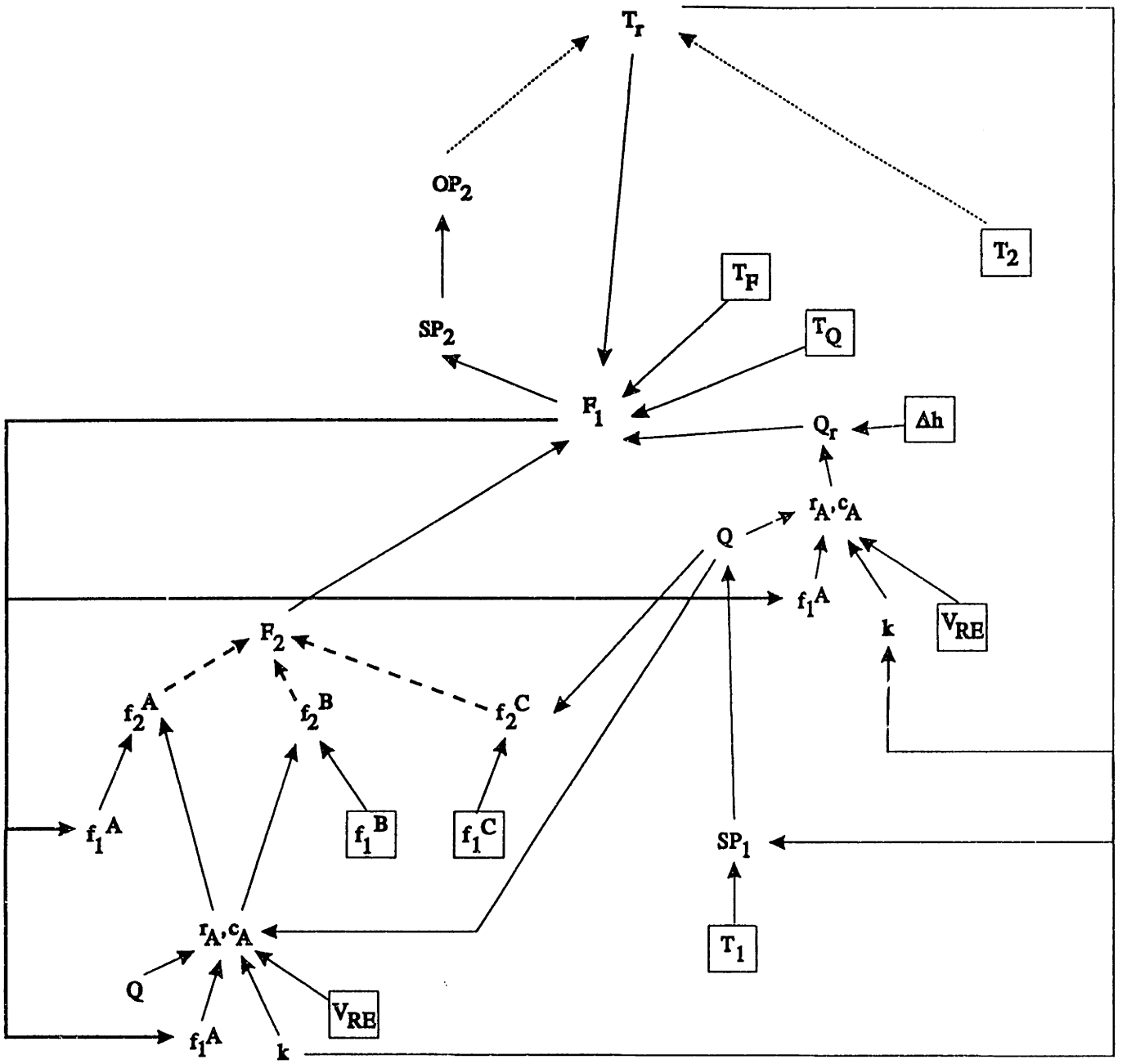


Figure 4-18: Variable influence diagram: assumption set 1



Key:
- boxed ≡ input variable
- unboxed ≡ output variable
- ·····▶ ≡ assumption-1 cause and effect
- – –▶ ≡ assumption-2 cause and effect
- ——▶ ≡ assumption-3 cause and effect
- ——▶ ≡ structural cause and effect

Figure 4-19: Variable influence diagram: assumption set 2

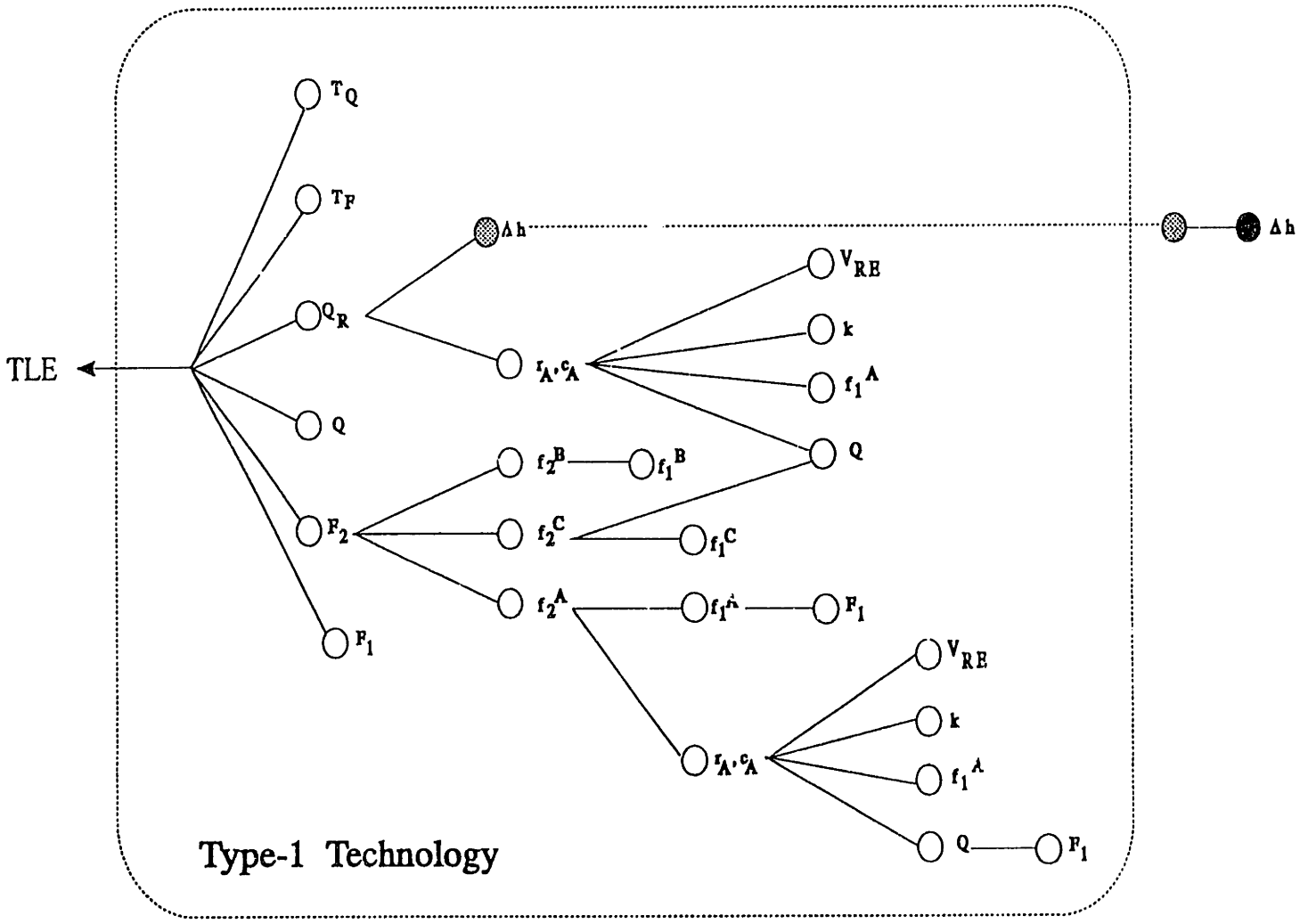


Figure 4-20: Technology Association: Type-1

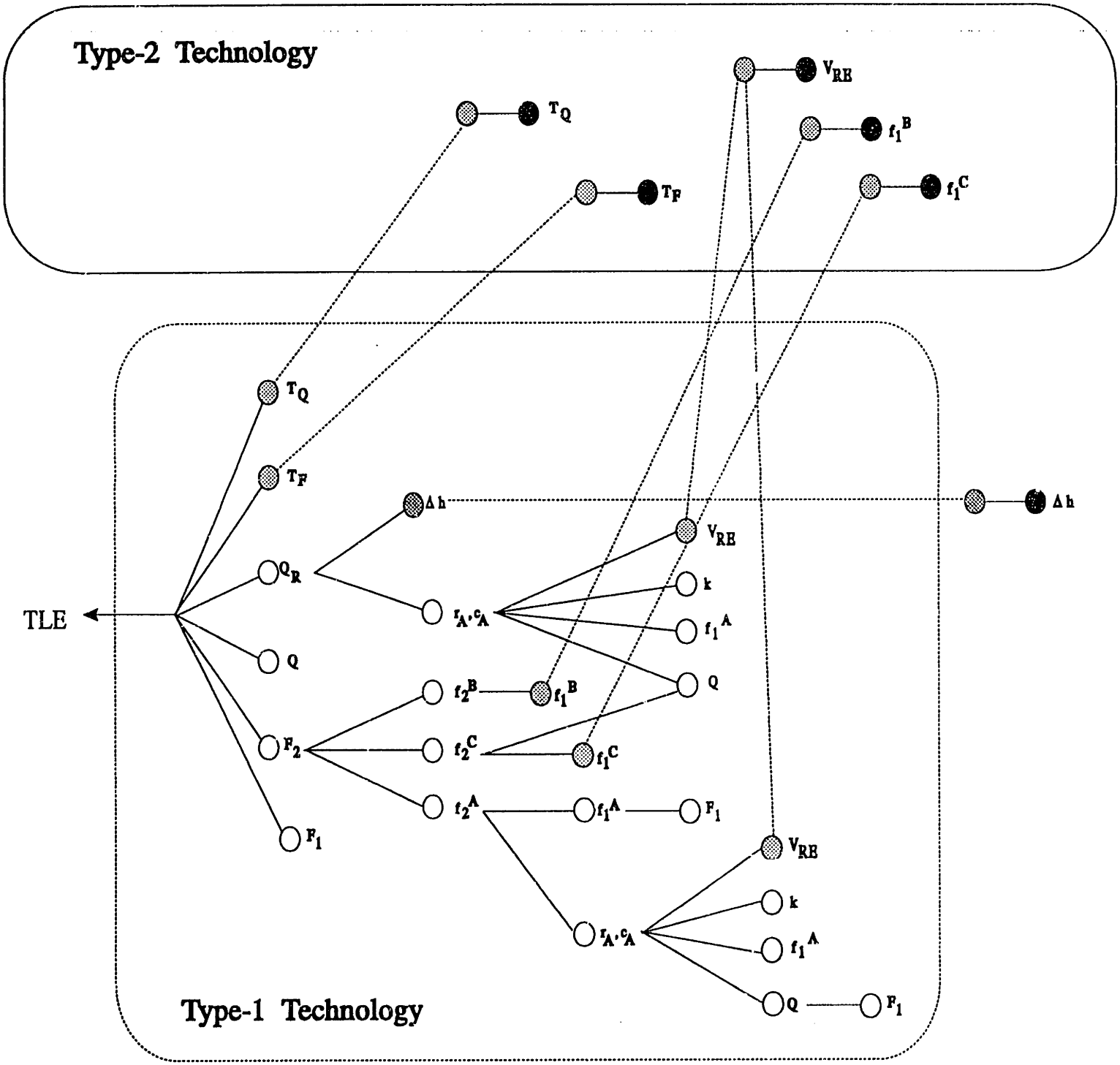


Figure 4-21: Technology Association: Type-2

described is autocatalytic. Trace amounts of product in the feed can catalyze the reaction leading to a thermal disturbance that ultimately could trigger a reaction runaway.

Lastly, Type-3 technologies are associated with the variable influence pathway. These technologies require active control to mitigate a disturbance and, consequently, are protective systems associated with the process flow sheet. The association of Type-3 technologies, to nodes enabling the top level event is shown in Figure 4-22. Notice that certain control actions cannot be modeled easily without over specifying the system. For example, the manual override of the quench valve by the operator. For this reason, the constraint list as described in Chapter 3 is associated with each piece of process equipment. This allows us to associate in an *a postori* manner additional control structures which are available to the process.

Figure 4-23 illustrates how the different technology types are used to identify potential root causes which can enable the top level event. This illustration makes explicit that disturbances, which enter into the system, can only enter through the inputs identified by Type-2 and Type-3 technologies. These disturbances can either be modeled or unmodeled. Modeled disturbances can only affect the process through its input variables and are therefore identified by the pathways leading to the top level event. Unmodeled disturbances can act in one of two ways: it can affect the value of a variable through an unmodeled relationship or it can change the causality described by the pathway, negating the assumption set from which it was built. However, since various assumption sets are used to construct the alternative pathways leading to the top level event, unmodeled disturbances which enable it can be minimized. Furthermore, due to the manner in which the top level event has been identified, it can only be enabled through the set of variables which describe the state immediately preceding it. Since any disturbance must eventually pass through that state, if it is to enable the top level event, control objectives

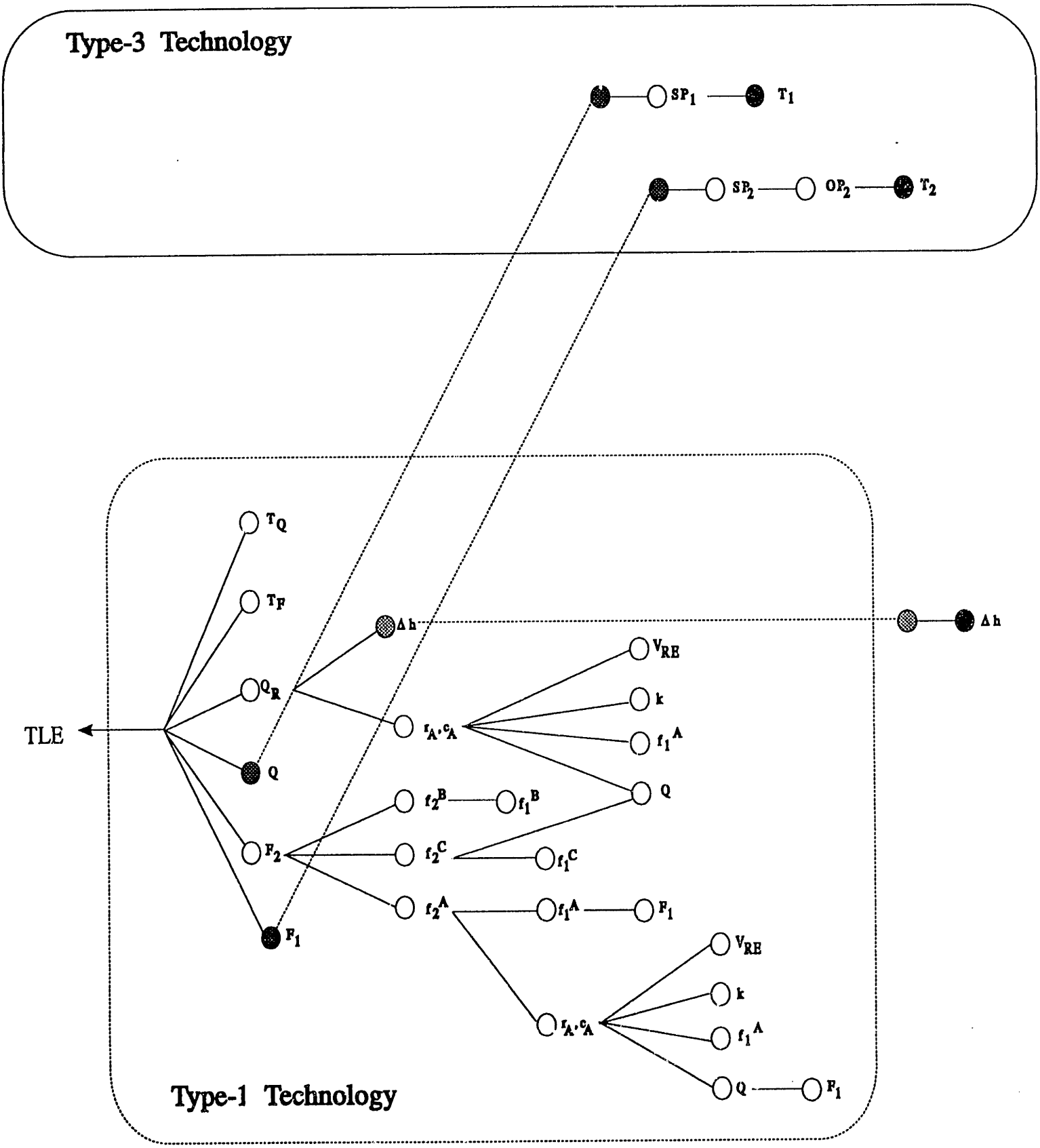


Figure 4-22: Technology Association: Type-3

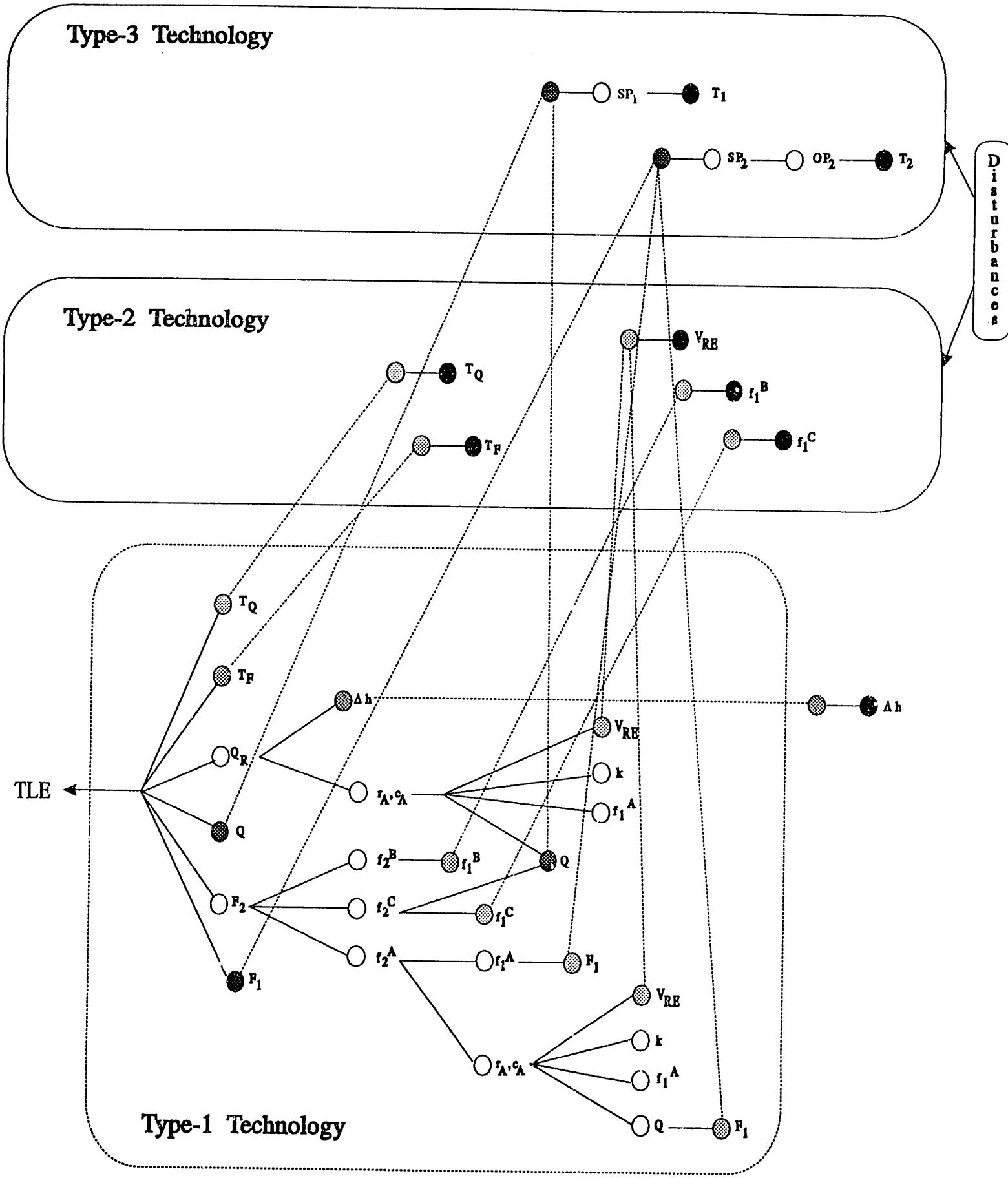


Figure 4-23: Root Cause Construction

designed around these variables can mitigate foreseen and unforeseen disturbances that lead to a top level event.

Using the knowledge associated with different technology types, and the pathways which lead to a top level event, a qualitative fault tree is constructed. This is illustrated in Figure 4-24. Note specifically the need for a special gate. This gate is required because without quantitative assessment it cannot be determined whether the gate is a single and-gate, a single or-gate, or a structure containing and-gates and or-gates.

The qualitative fault tree shown in Figure 4-24 suggests that the top level event can be enabled through a change in the input of T_Q (e.g. this could imply no quench). Using the algorithm presented in Section 4.2.2, the logical gate constructed from this input variable is an and-gate: the result of a controlled variable involving an unbranched node. T_F, feed temperature, is the next variable with a pathway to a top level event. Similarly, an and-gate is constructed for feed temperature, T_F, representing a thermal deviation demand in the feed and the failure of the operator to notice the upset or take the required action. Quench flow rate, Q, also has a pathway to the top level event (see Figure 4-22). Following the algorithm proposed, construction of logical gates associated with this pathway illustrates that a demand by Q can be enabled in one of two ways: the stem position of valve 1 fails to obtain the necessary position (i.e. the valve fails to close) or the sensor setpoint T_1 is in error. The resulting qualitative fault tree is shown in Figure 4-24.

Unlike T_2, T_F, and Q, the outlet feed, F_2, requires a special gate to model the logical consequence of root causes passing through it. This gate takes as its input f_2^B, f_2^C, and f_2^A. Figure 4-23 illustrates that f_2^B obtains its input directly from f_1^B, a Type-2 technology. Consequently, any disturbance in f_1^B could potentially enable F_2. Since f_2^C and f_2^A are branched nodes, they in turn require construction of special-gates. The

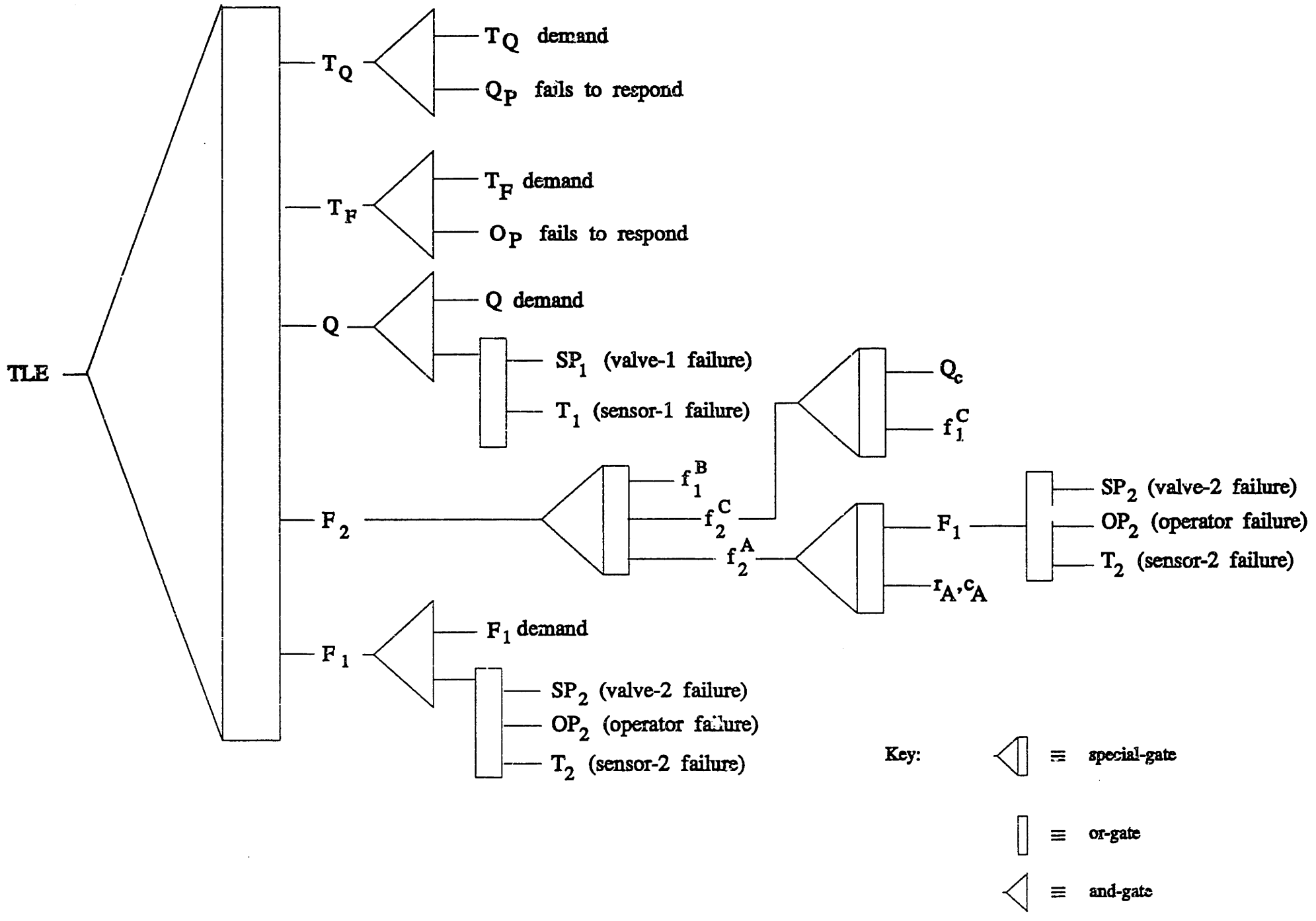


Figure 4-24: Qualitative Fault Tree: Assumption Set 1

special-gate constructed from f_2^C takes as its input Q, and f_1^C. Similar to f_1^B, f_1^C is a Type-2 technology which takes its input directly from the external world. Therefore, any disturbance in the external surroundings which can lead to a change in the value of f_1^C could potentially also enable the top level event.

Variable f_2^A being a branched node, also requires the construction of a special gate. This gate requires as its input F_1 and $\{r_A, C_A\}$. Although the expansion for this node is not shown, what is important to recognize that with the exception of V_{RE}, each of the inputs leading to this set have been covered previously. The implications of changing V_{RE}: a Type-2 technology; is that a change in reactor volume could enable a top level event. Although the mechanism for this process has not been modeled, is could be the result of an improper design or the accumulation of material internal to the reactor. What is important is that the effect of V_{RE} has been made explicit.

The final variable with a pathway leading to the top level event is F_1: an unbranched node. Since F_1 has associated with it a Type-3 technology as its protective system. A demand on F_1 can be mitigated by the proper positioning of the stem controlling valve 2, proper action by the operator, and a proper setpoint (i.e. T_2) on the temperature controller. The qualitative fault tree constructed for this sequence is shown in Figure 4-24.

A condensed version of the qualitative fault tree shown in Figure 4-24, involving only Type-3 technologies is shown in Figure 4-25. It identifies two principal pathways to the top level event: Q and F_1. Notice that a special-gate is still needed to connect these inputs to the output, T_R. The necessity of the special-gate will remain until quantitative analysis can determine whether the top level event can be enabled by either Q or F_1, or whether both are required. This depends on the dynamics of the system: reaction rates,

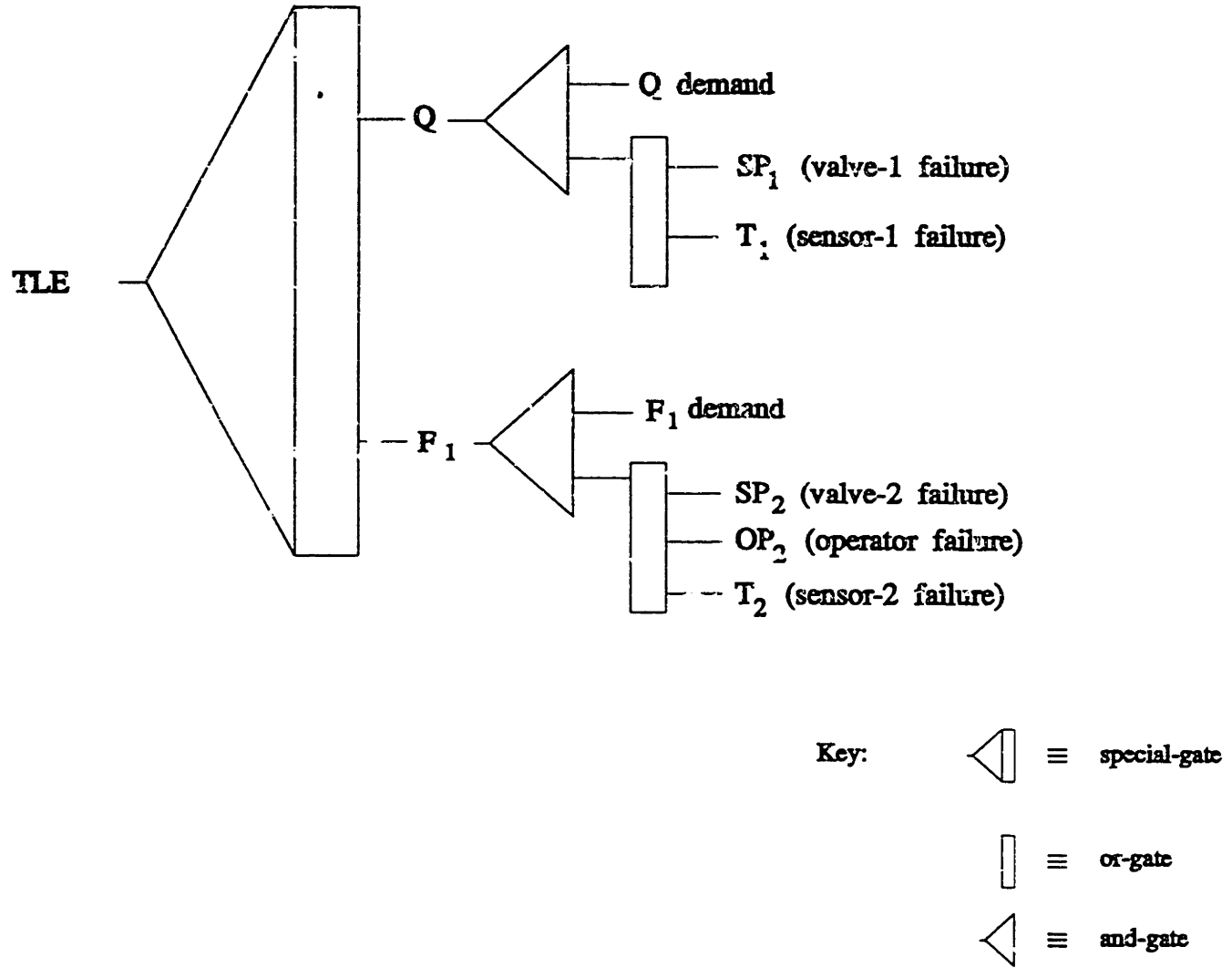


Figure 4-25: Qualitative Fault Tree: Type-3 Technology (Assumption Set 1)

time requirements for runaway, heat release rates, heat removal rates, reactor volume, quench rates, etc.

The fault tree cited in literature for this process is shown in Figure 4-26 [Battelle, 1985]. Notice the similarity between Figure 4-26 and Figure 4-25; particularly in the structure of the two trees, and recognize that as a result of quantitative analysis Figure 4-26 has an and-gate as its top level gate. More importantly, recognize that without complete quantification of the root causes the fault tree given in Figure 4-26 may be incomplete. This occurs because the pathways leading to the top level event have not been deductively established and made explicit.

4.3.2 Case Study 2: formaldehyde oxide process

Consider the process flowsheet show in Figure 4-27 for the production of formaldehyde. The process consists of a vertical packed bed catalytic reactor which is equipped with temperature and pressure sensors, an explosion vent, and a distributor plate. The reacting gas mixture flows downward through the tubes and transfers the heat of reaction to a circulating heat transfer medium, on the shell side of the reactor. The heat transfer medium in turn vaporizes feed water to produce high pressure steam. System pressure is maintained by controlling the rate of feed water introduction. Exit gases, from the reactor, pass through a heat recovery exchanger, where low pressure steam is generated, and then to an absorption column where water is used as the scrubbing medium. A substantial portion of the absorber overhead gas is recycled back to the feed system.

The design intent is to maximize the production of formaldehyde, through the conversion of methanol and air, without entering the flammability envelope. For this purpose, gaseous reactor effluents (e.g. nitrogen) are recycled. Pure methanol is pumped into the process at a constant rate, specified by the setpoint (SP_3), and vaporized. Downstream methanol concentrations are regulated by the amount of air and recycle introduced into

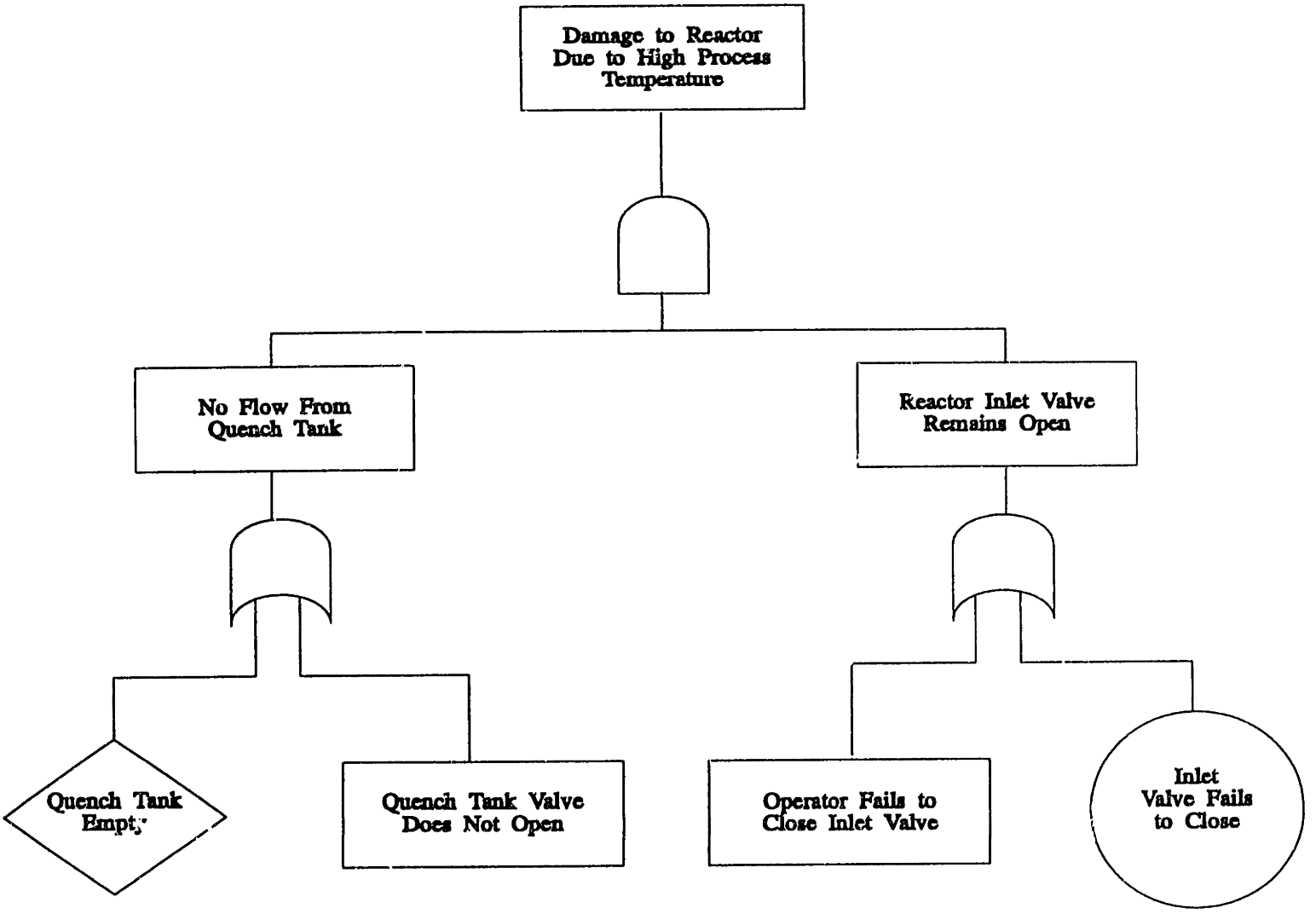


Figure 4-26: Cited Fault Tree

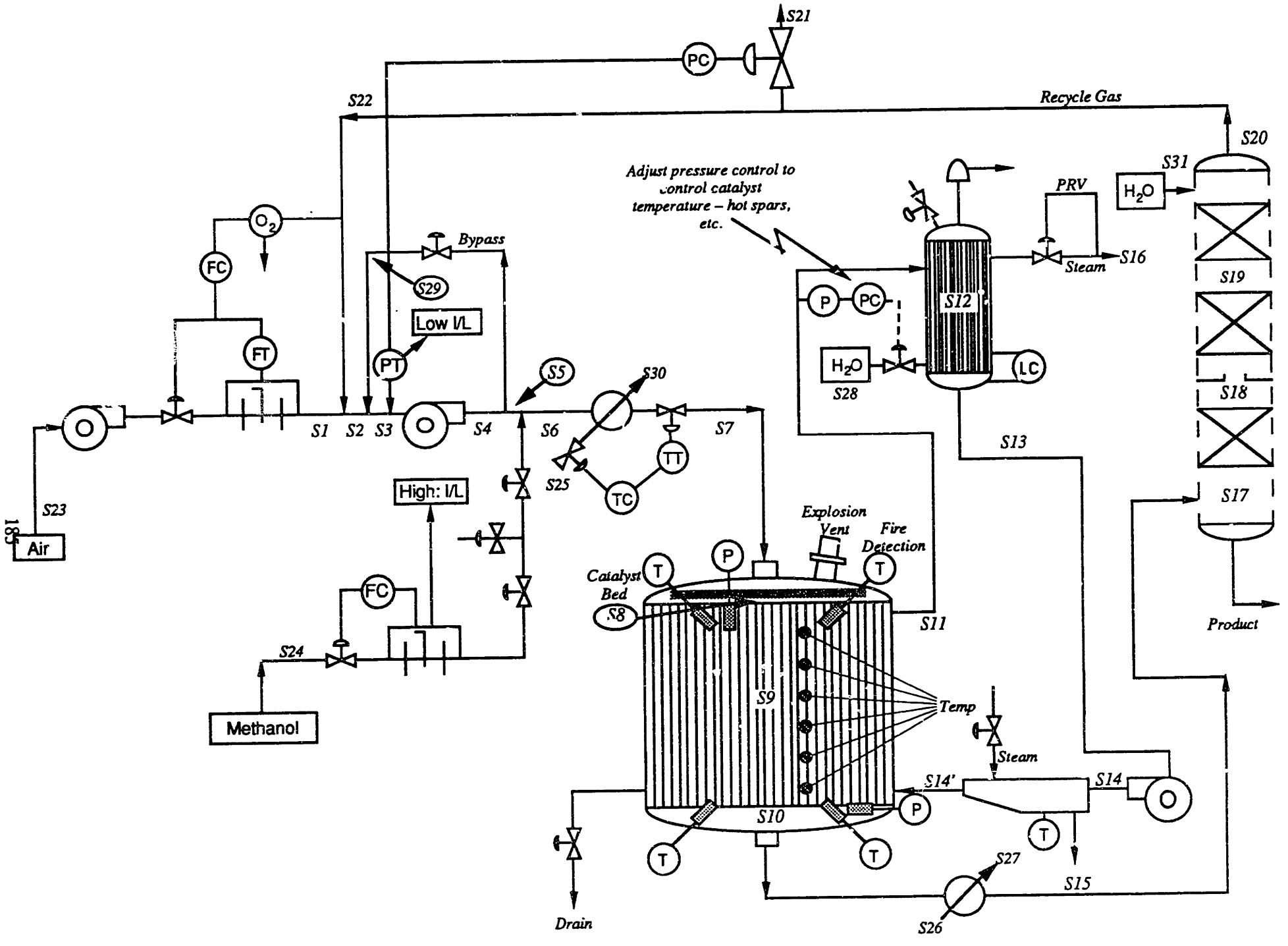


Figure 4-27: Formaldehyde Oxide Process

the system. An oxygen analyzer on the recycle system measures the oxygen content and adjusts the introduction of air into the system so as to maintain an oxygen level less than setpoint (SP_1). The flow of S_3, is maintained by controlling the amount of recycle vented based on SP_2.

The analysis for the deductive determination of root causes begins by constructing the set of process equations which describe the process flowsheet. Assuming that the TLE of interest is an ignitable mixture at the reactor inlet, our focus is on the set of equations that describe the mass balance of the system. [We ignore the system energy balance because the reactor temperature exceeds the autoignition temperature for the mixture.] Equations describing the process are shown in Figure 4-28a-d; therein: "S" denotes total molar feed rate; "y" denotes mole fraction; "r_A" is a rate expression; "V" is the volume of the reactor; "S_A" denotes catalyst surface area; "T" denotes temperature; "VP" denotes valve stem position; and "SP" denotes controller setpoint. Complex relationships (e.g. control equations and rate expressions) are illustrated in Figure 4-28 showing only functional dependencies. The structural matrix resulting from this set of equations is given in Figure 4-29.

Construction of the various variable influence pathways begins with the identification of input variables. Following the procedures outlined in Section 4.2.1, input set assignment is initiated. Input specification is ameliorated using qualified assumptions. Figure 4-30 illustrates boundaries of qualified Assumption-1. By focusing our attention on the reactor, boundaries of the system identify reactor volume, V. and reactor temperature (because it is controlled externally) as input variables. By expanding the boundaries of the system, qualified Assumption-2 (see Figure 4-31) identifies the invariant intensive properties describing the feeds of air and methanol as input variables. These are y^o_{23} and y^a_{24}. Step 3 of the input assignment procedure assigns the setpoints: SP_1, SP_2, and SP_3; as input variables. The boundary of leading to this specification (i.e. Assumption-3) is

Flow Balance

$$
\begin{aligned}
S_1 + S_{22} &= S_2 \\
S_2 &= S_3 \\
S_3 &= S_4 \\
S_4 &= S_5 \\
S_5 + S_{24} &= S_6 \\
S_6 &= S_7 \\
S_7 &= S_8 \\
S_8 &= S_{10} \\
S_{10} &= S_{15} \\
S_{15} &= S_{17} + S_{20} \\
S_{20} &= S_{21} + S_{22} \\
S_{23} &= S_1
\end{aligned}
$$

Control Equations

$$
\begin{aligned}
S_1 &= f\,(y^o_{22}, SP_1) \\
VP_1 &= f\,(y^o_{22}, SP_1)
\end{aligned}
$$

$$
\begin{aligned}
S_{21} &= f\,(VP_2) \\
VP_2 &= f\,(S_3, SP_2)
\end{aligned}
$$

$$
\begin{aligned}
S_{24} &= f\,(VP_3) \\
VP_3 &= f\,(y^a_6, SP_3)
\end{aligned}
$$

Figure 4-28 (a): Process Equations

Continuity Equations

$$
\begin{aligned}
y^o_1 + y^n_1 &= 1 \\
y^o_2 + y^n_2 + y^a_2 + y^f_2 &= 1 \\
y^o_3 + y^n_3 + y^a_3 + y^f_3 &= 1 \\
y^o_4 + y^n_4 + y^a_4 + y^f_4 &= 1 \\
y^o_5 + y^n_5 + y^a_5 + y^f_5 &= 1 \\
y^o_6 + y^n_6 + y^a_6 + y^f_6 &= 1 \\
y^o_7 + y^n_7 + y^a_7 + y^f_7 &= 1 \\
y^o_8 + y^n_8 + y^a_8 + y^f_8 &= 1 \\
y^o_{10} + y^n_{10} + y^a_{10} + y^f_{10} &= 1 \\
y^o_{15} + y^n_{15} + y^a_{15} + y^f_{15} &= 1 \\
y^a_{17} + y^f_{17} &= 1 \\
y^o_{20} + y^n_{20} + y^a_{20} + y^f_{20} &= 1 \\
y^o_{21} + y^n_{21} + y^a_{21} + y^f_{21} &= 1 \\
y^o_{22} + y^n_{22} + y^a_{22} &= 1 \\
y^o_{23} + y^n_{23} &= 1
\end{aligned}
$$

Figure 4-28 (b): Process Equations

Component Balances

$$
\begin{aligned}
y^{o}_{23} &= y^{o}_{1} \\
y^{o}_{1}S_1 + y^{o}_{22}S_{22} &= y^{o}_{2}S_2 \\
y^{a}_{22}S_{22} &= y^{a}_{2}S_2 \\
y^{f}_{22}S_{22} &= y^{f}_{2}S_2 \\
y^{o}_{2} &= y^{o}_{3} \\
y^{a}_{2} &= y^{a}_{3} \\
y^{f}_{2} &= y^{f}_{3} \\
y^{o}_{3} &= y^{o}_{4} \\
y^{a}_{3} &= y^{a}_{4} \\
y^{f}_{3} &= y^{f}_{4} \\
y^{o}_{4} &= y^{o}_{5} \\
y^{a}_{4} &= y^{a}_{5} \\
y^{f}_{4} &= y^{f}_{5} \\
y^{o}_{5}S_5 &= y^{o}_{6}S_6 \\
y^{a}_{5}S_5 + y^{a}_{24}S_{24} &= y^{a}_{6}S_6 \\
y^{f}_{5}S_5 &= y^{f}_{6}S_6 \\
y^{o}_{6} &= y^{o}_{7} \\
y^{a}_{6} &= y^{a}_{7} \\
y^{f}_{6} &= y^{f}_{7} \\
y^{o}_{7} &= y^{o}_{8} \\
y^{a}_{7} &= y^{a}_{8} \\
y^{f}_{7} &= y^{f}_{8} \\
y^{o}_{8} &= y^{o}_{10} - R_A V \\
y^{a}_{8} &= y^{a}_{10} - R_A V \\
y^{f}_{8} &= y^{f}_{10} + R_A V \\
R_A &= k_L S_A y^{o}_{8} y^{a}_{8} S_8 \\
k_L &= f(T_R)
\end{aligned}
$$

Figure 4-28 (c): Process Equations

Component Balances (cont'd)

$$
\begin{aligned}
y^{o}_{10} &= y^{o}_{15} \\
y^{a}_{10} &= y^{a}_{15} \\
y^{f}_{10} &= y^{f}_{15} \\
y^{o}_{15}S_{15} &= y^{o}_{20}S_{20} \\
y^{a}_{15}S_{15} &= y^{a}_{17}S_{17} + y^{a}_{20}S_{20} \\
y^{f}_{15}S_{15} &= y^{f}_{17}S_{17} + y^{f}_{20}S_{20} \\
y^{a}_{17} &= f\,(y^{a}_{15}) \\
y^{o}_{20} &= y^{o}_{21} \\
y^{o}_{20} &= y^{o}_{22} \\
y^{a}_{20} &= y^{a}_{21} \\
y^{a}_{20} &= y^{a}_{22} \\
y^{f}_{20} &= y^{f}_{21} \\
y^{f}_{20} &= y^{f}_{22}
\end{aligned}
$$

Figure 4-28 (d): Process Equations

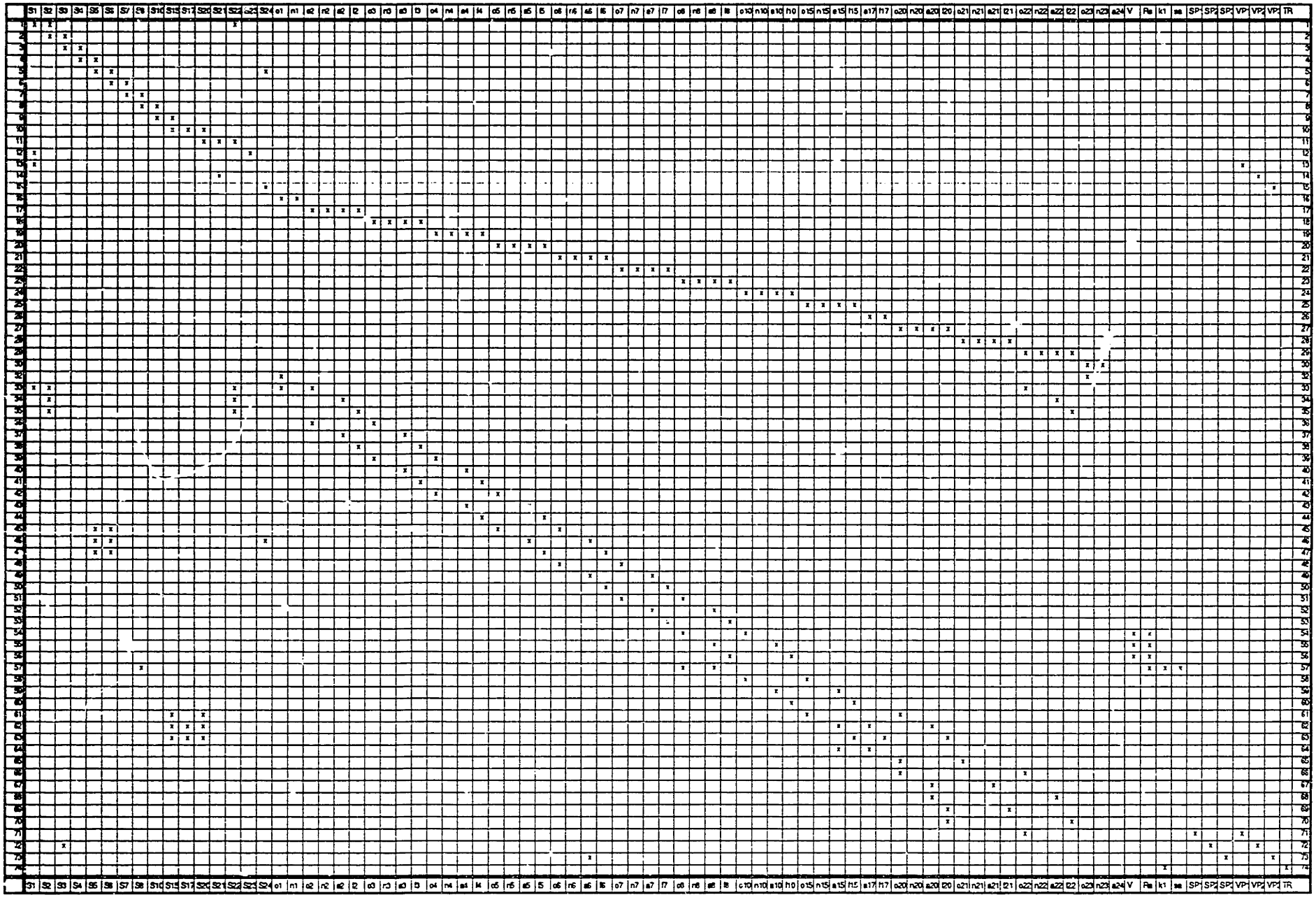


Figure 4-29: Structural Matrix

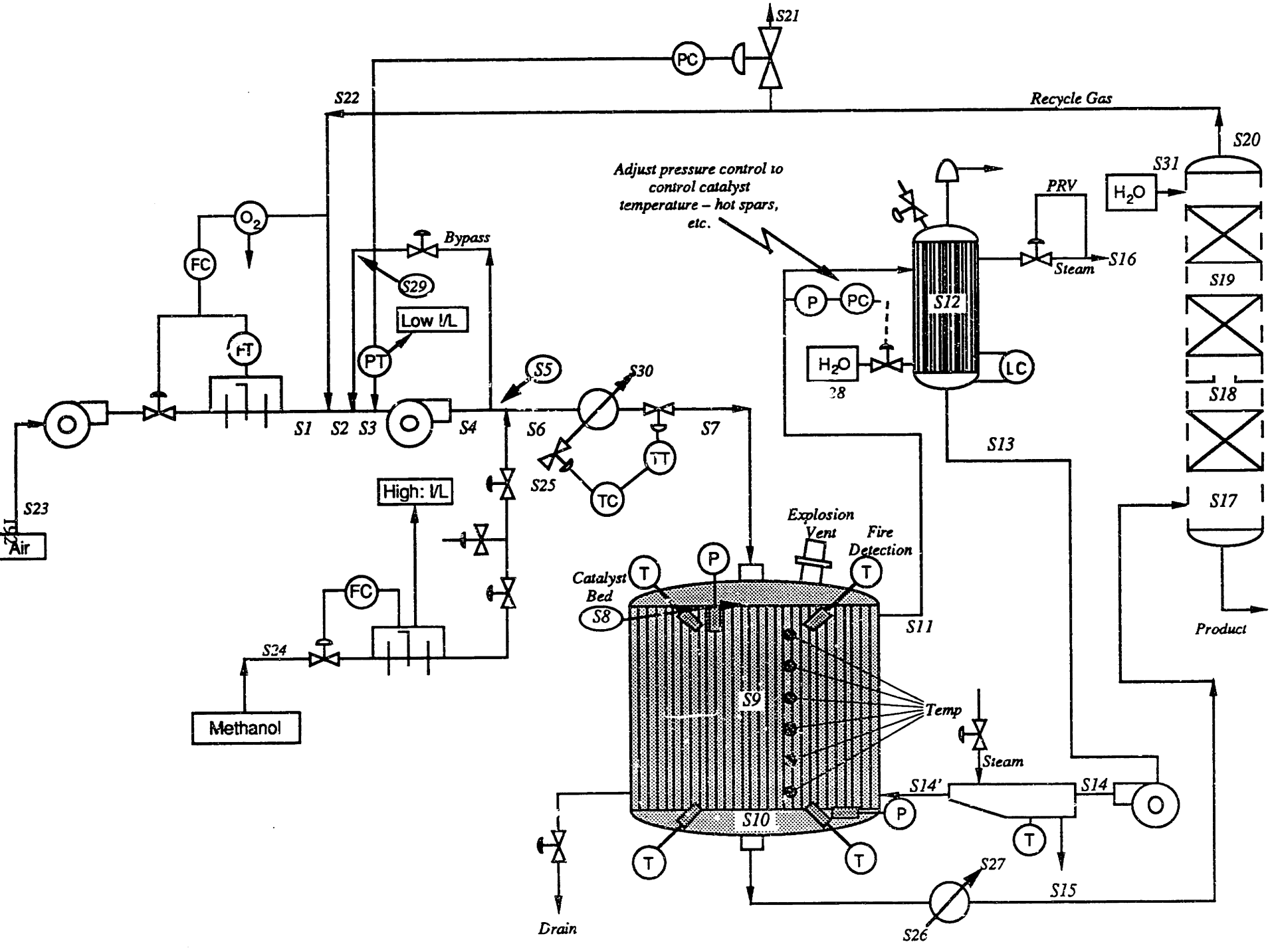


Figure 4-30: Input specification: qualified assumption-1

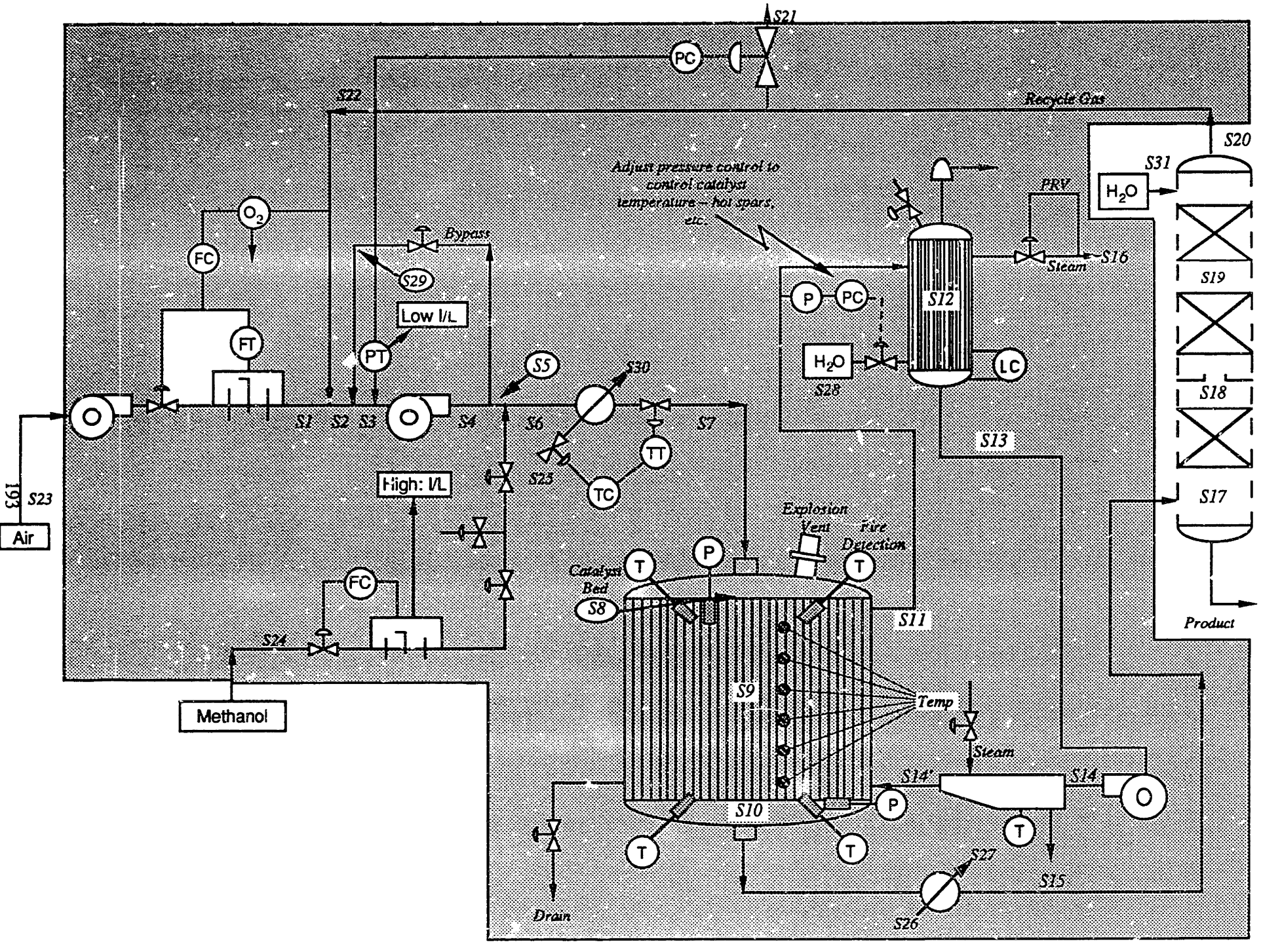


Figure 4-31: Input specification: qualified assumption-2

shown in Figure 4-32. Figure 4-33 illustrates the structural matrix resulting from this assignment.

Once inputs are assigned, output assignment begins as described in Section 4.2.2. Step 2.2 of this procedure identifies S_{23} as taking its output from equation [12]. Similarly, Step 2.2 identifies the mole fractions: y^n_{1} to y^n_{15}; and y^n_{20} to y^n_{23} as obtaining their assignments from equations: [16]-[25] and [27]-[30], respectively. Step 2.4 assigns **k** as the output of equation [73].

Elimination of the assigned rows and columns results in no direct causality being identified (i.e. there is no unique assignment). To break this loop we make a tentative assumption that S_1 obtains its output from equation [13]. This implies that the flow, S_1, responds to valve position (VP_1) rather than the other way around. This assumption is then catalogued as Assumption-1. Retraction of this assumption allows the causality of the set of process equations to be modified. The assignment of equation [13] allows the assignment of additional outputs but again a tentative assumption is necessary to break the loop. Assumption-2 and Assumption-3 provide an interpretation of control loop causality. These assignments specify that S_{21} obtains its output from equation [14] and y^a_6 obtains its output from equation [72]. In both equations it is assumed that valve position, VP_2 and VP_3, are independent variables . These assumptions are then catalogued. By continuing the assignment process, the outputs shown in Figure 4-34 results. Notice that variables y^a_{20} and S_A have not been assigned. These variables are forced to become inputs. Final assignments are shown in Figure 4-35.

Descriptions of potentially hazardous states preceding the top level event are shown in Figure 4-36. These states make explicit variables and the values necessary to achieve the top level event. By constructing the variable influence diagram of y^o_8 and y^a_8, pathways leading to the top level event and the enabling underlying root causes can be

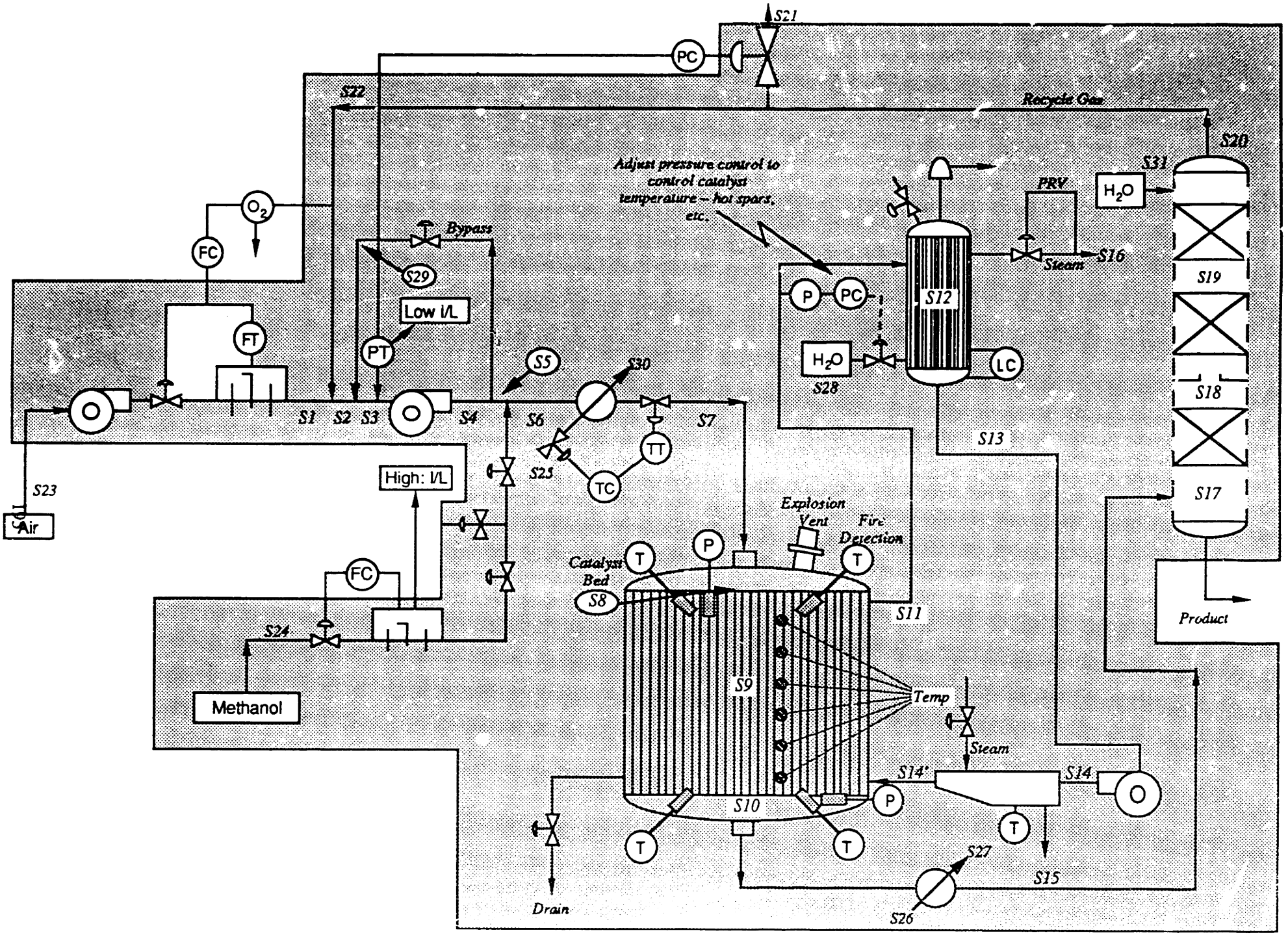


Figure 4-32: Input specification: qualified assumption-3

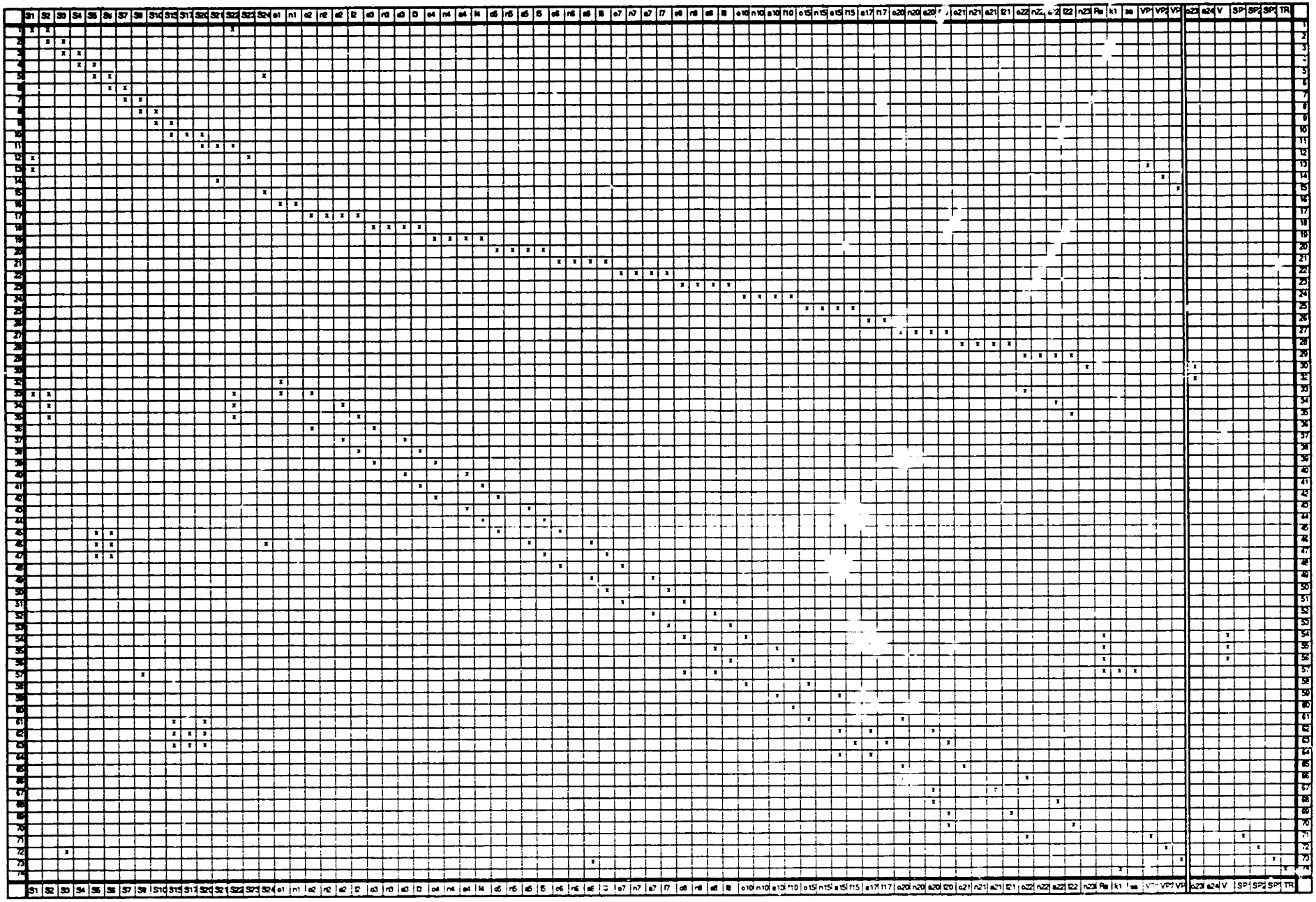


Figure 4-33: Structural Matrix: input assignment

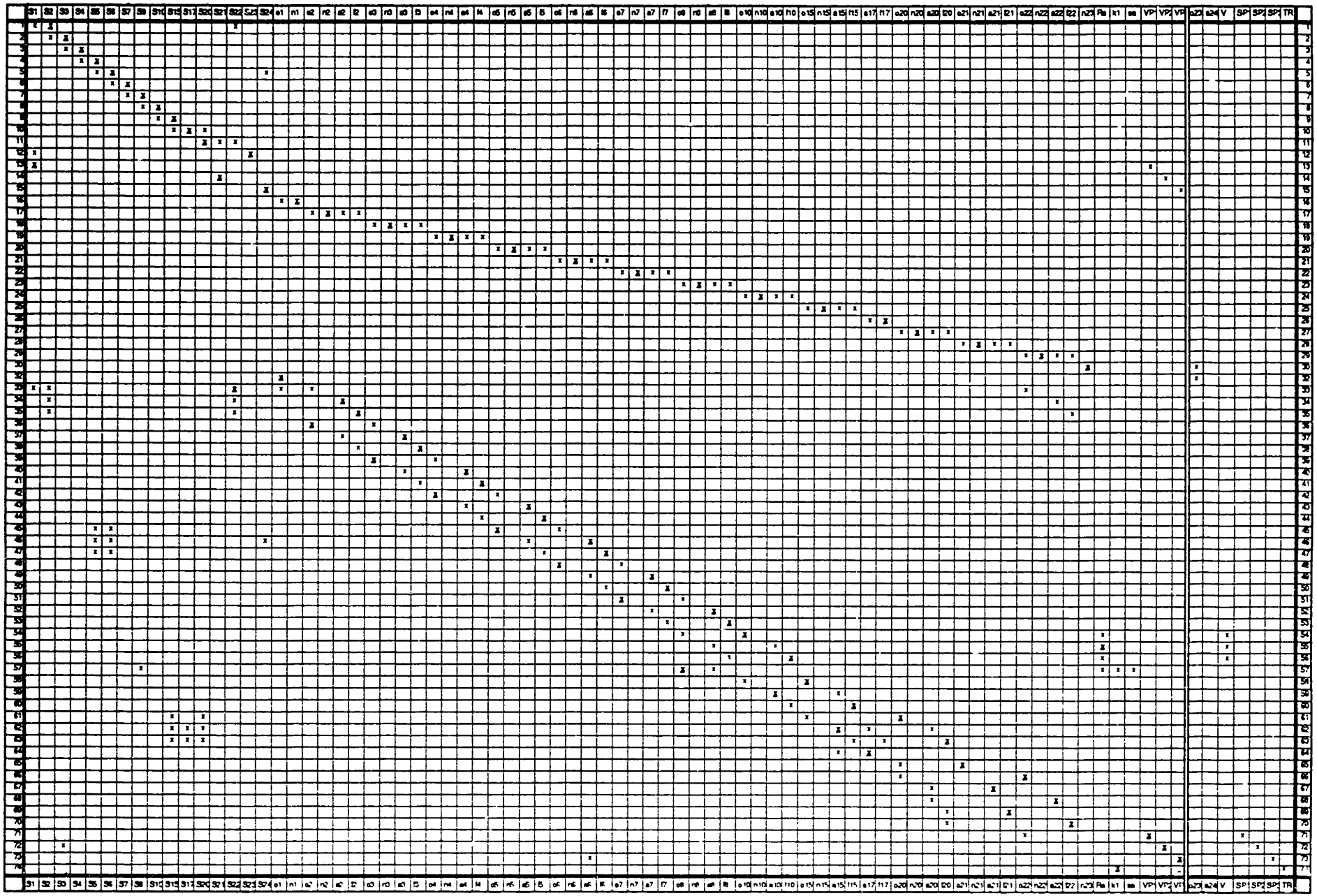


Figure 4-34: Structural Matrix: output assignment

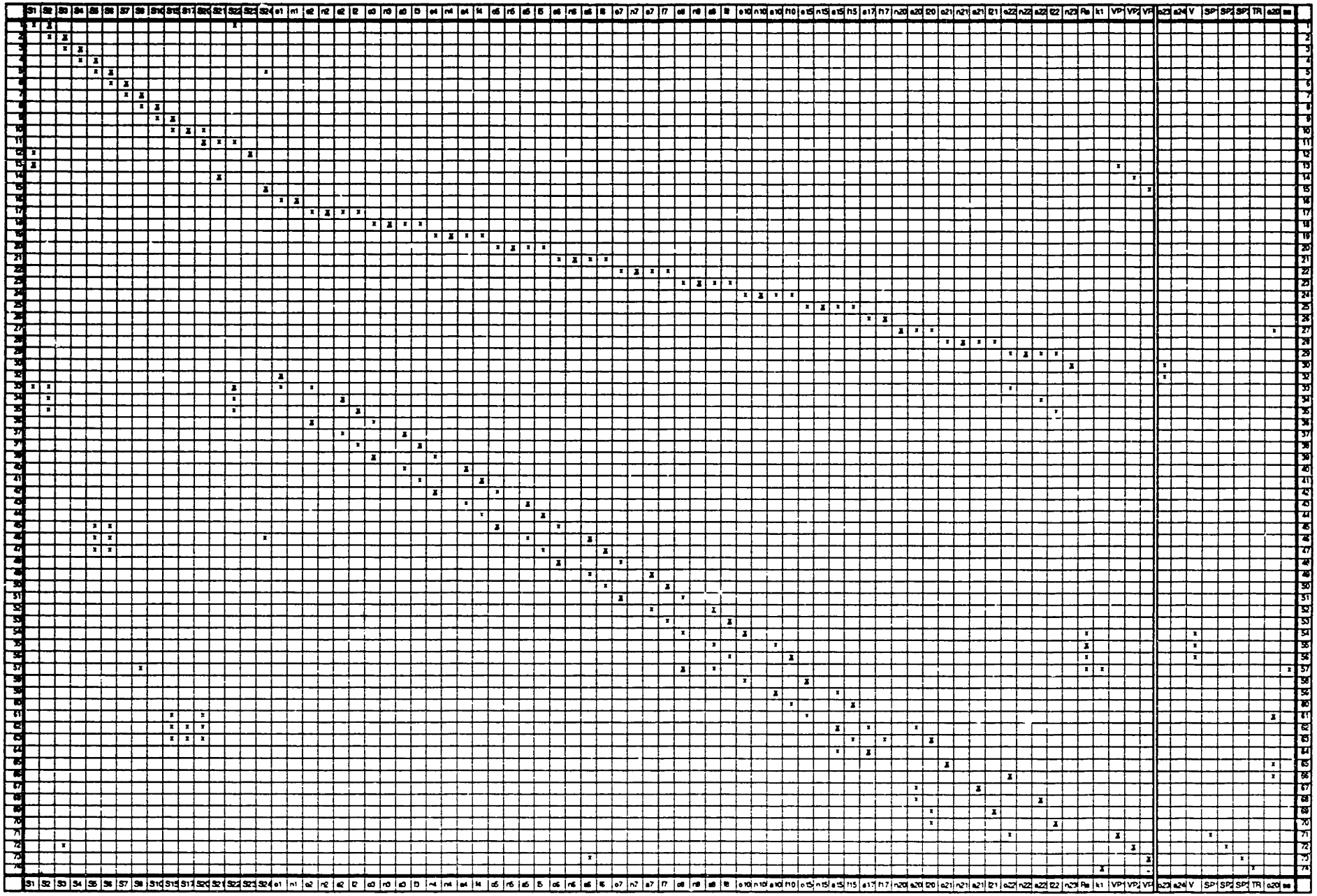


Figure 4-35: Structural Matrix: forced assignments

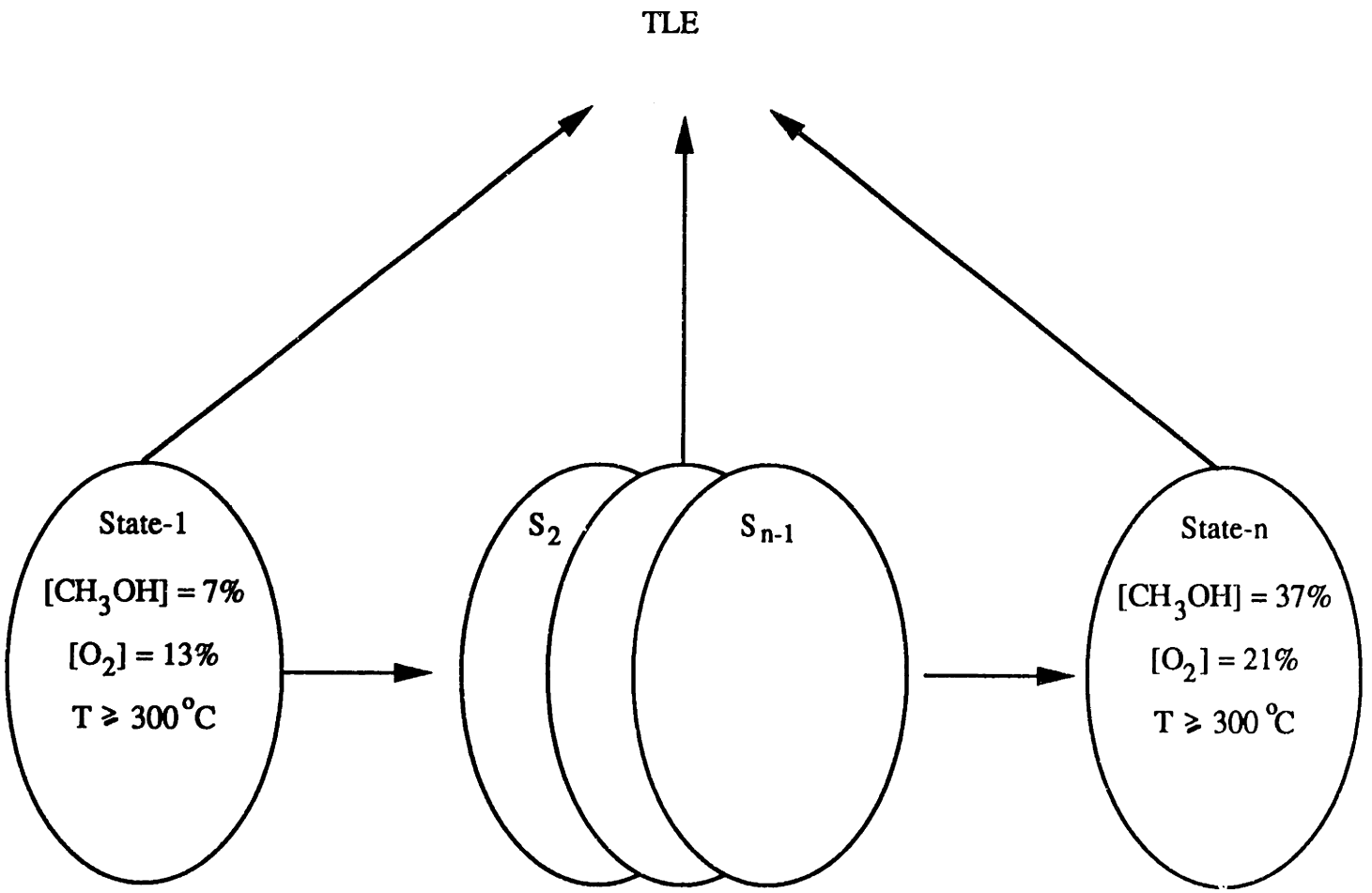


Figure 4-36: Hazardous state description

identified. A variable influence diagram for T_R is unnecessary because conditions within the reactor satisfy the value required (i.e. $T \geq 300°C$). It is established by the reactor's node description (see Chapter 3).

The variable influence diagram, resulting from this assumption set (i.e. Assumption-1, Assumption-2, and Assumption-3), is shown in Figure 4-37 for y^o_8. Similarly, Figure 4-38 illustrates the variable influence diagram for y^a_8. These diagrams in combination make explicit the pathways leading to the top level event by illustrating how disturbances can propagate through the network of equations.

By associating technology types with the variables contained in these diagrams the pathway of root causes is constructed. Figure 4-39 illustrates this association for the variable influence diagram described by assumption set 1 and leading to y^o_8. Notice. that the inputs to the system are entirely through Type-2 and Type-3 technologies. This implies that foreseeable disturbances must propagate through Type-2 and Type-3 technologies exclusively. Consequently, these technologies and the systems described by them will become the underlying root causes that enable the top level event. Furthermore, the variable influence diagram for y^o_8 makes explicit that TLE prevention necessitates the placement of control objectives on every variable which immediately precedes it: S_A, k_t, R_A, y^a_8, and S_8. Similarly, Figure 4-40 shows the variable influence diagram for y^a_8 with the association of various technology types to the variables contained in that diagram. Recognize that y^a_8 is contained in the pathway leading to y^o_8. As a consequence, Type-2 and Type-3 technologies associated with y^o_8 contain the set of root causes capable of enabling it.

By associating the enabling set of root causes for y^a_8, y^o_8, and T_R, root causes which lead to the TLE can be constructed. This is shown in Figure 4-41. Since y^o_8 takes its output from an equation containing y^a_8, construction of the root causes that enable the

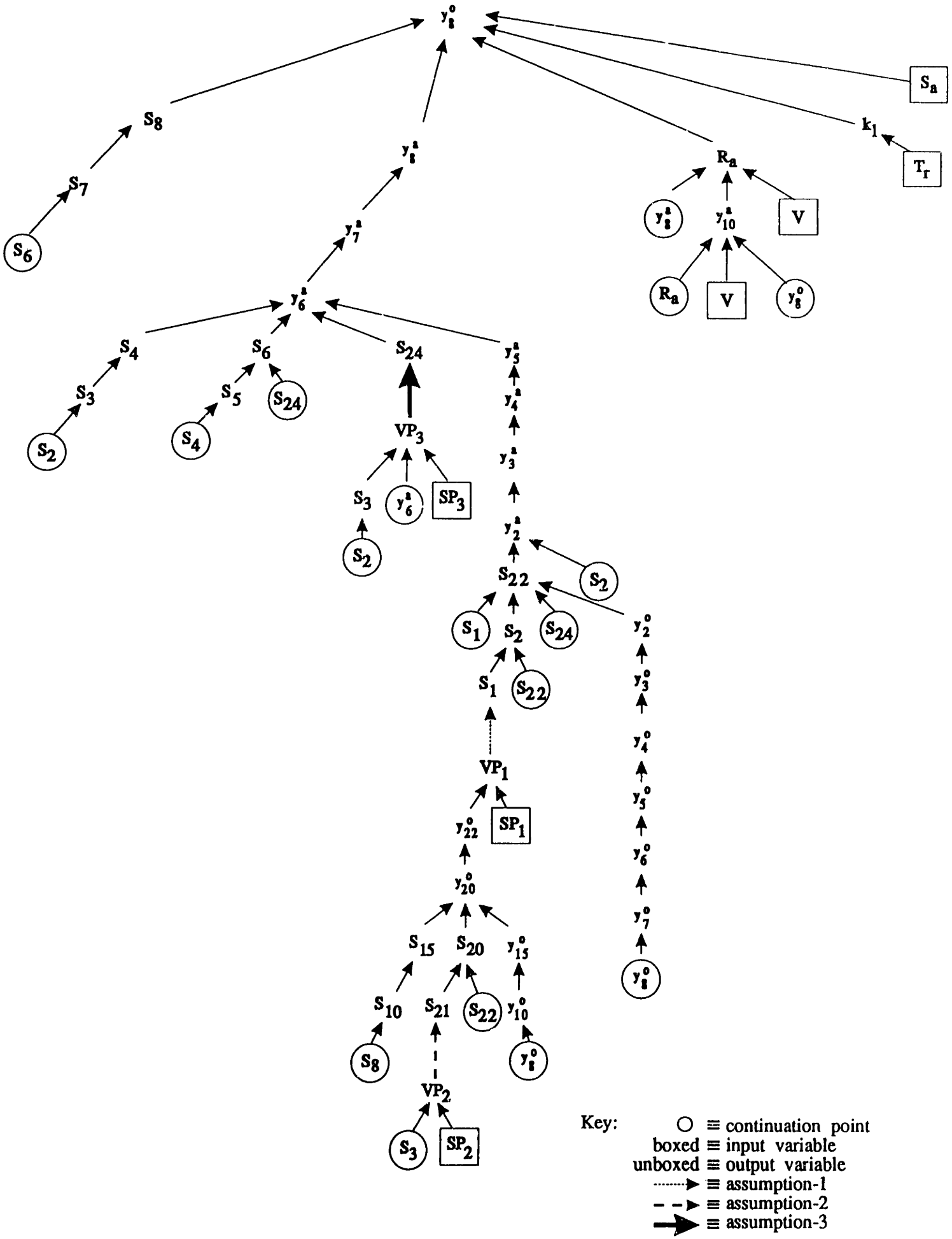


Figure 4-37: Variable influence diagram: $y_8^{o\,201}$

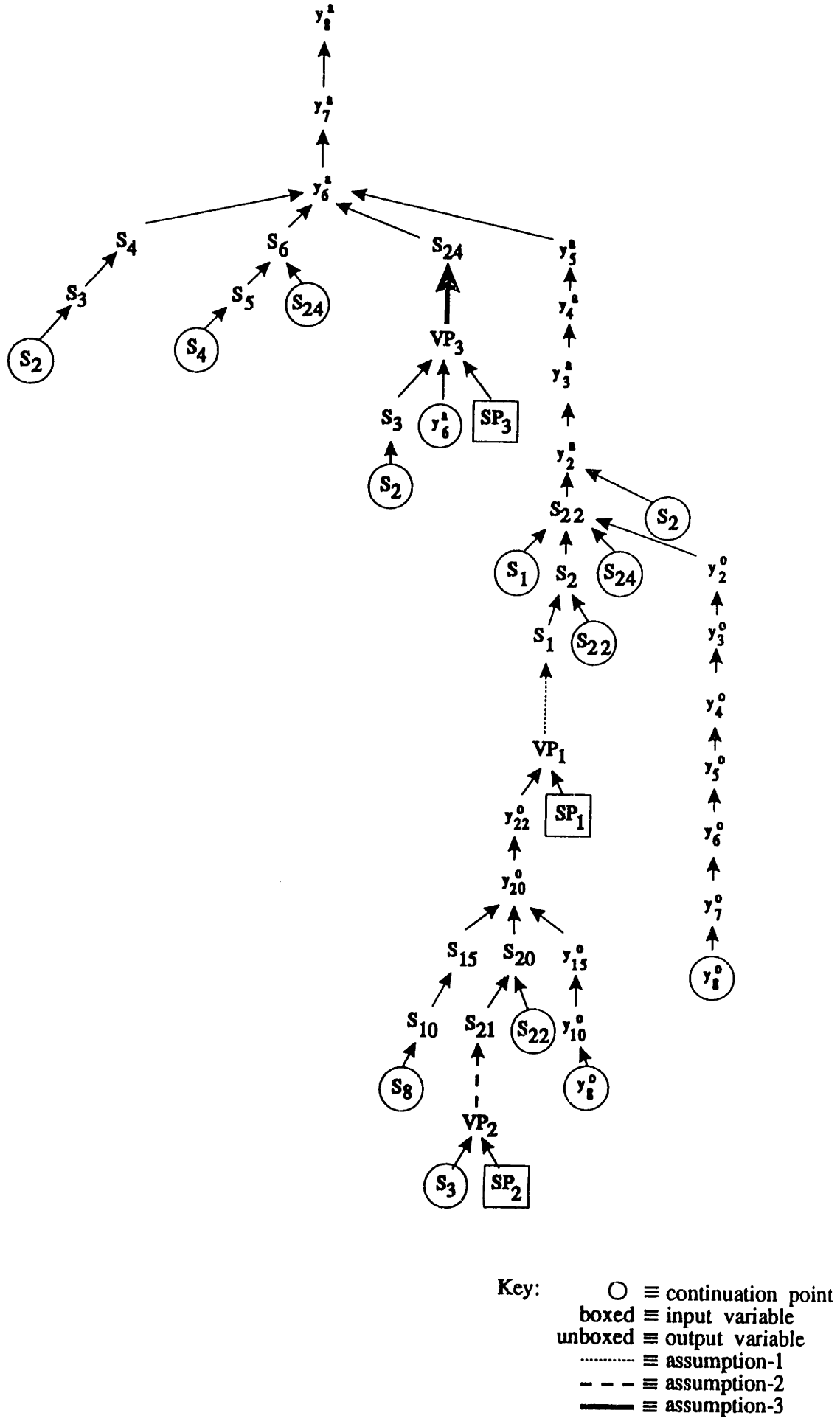
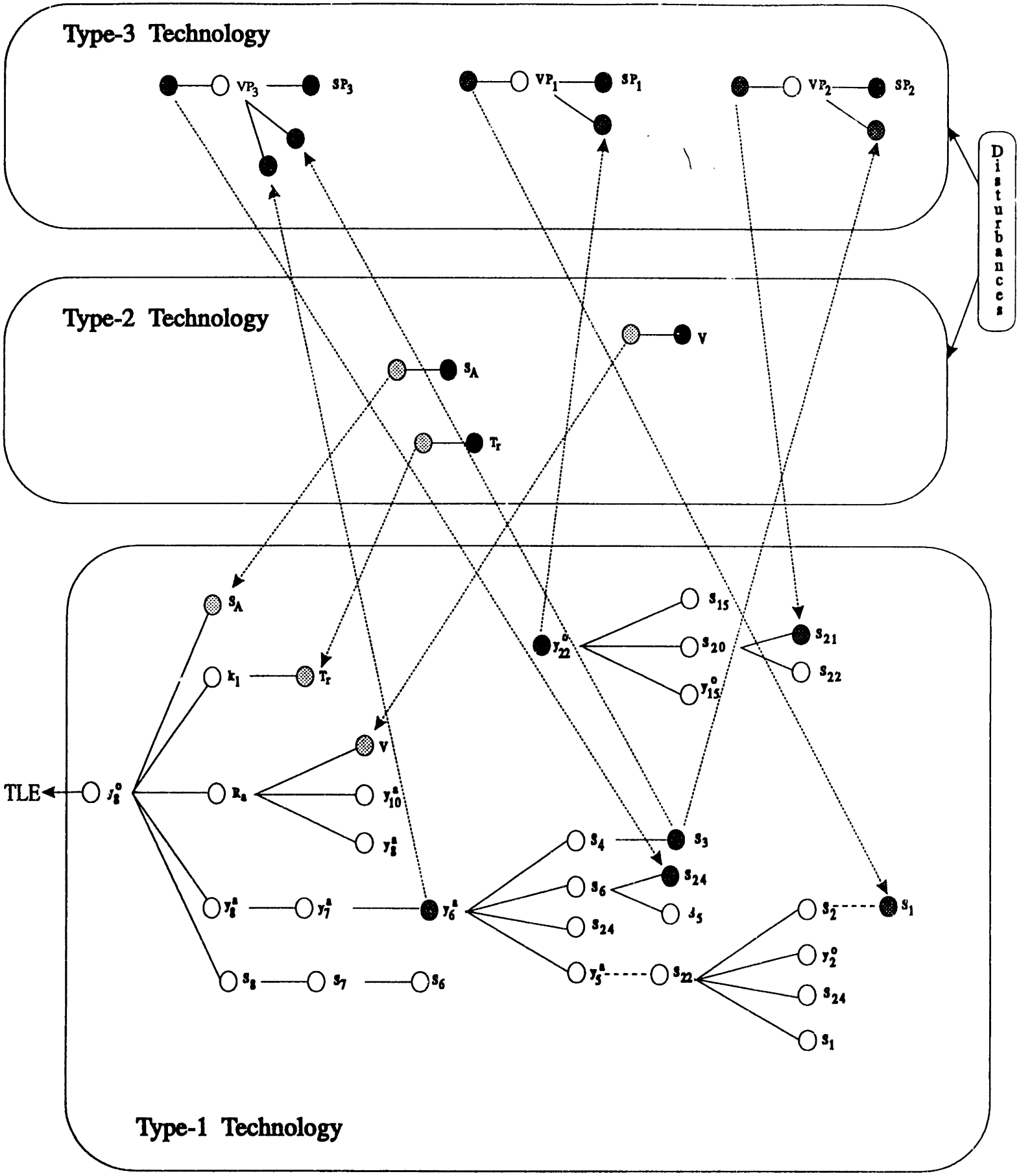


Figure 4-38: Variable influence diagram: y_8^a



Key: ------ abbreviated branch

Figure 4-39: Root cause construction: y_8^o

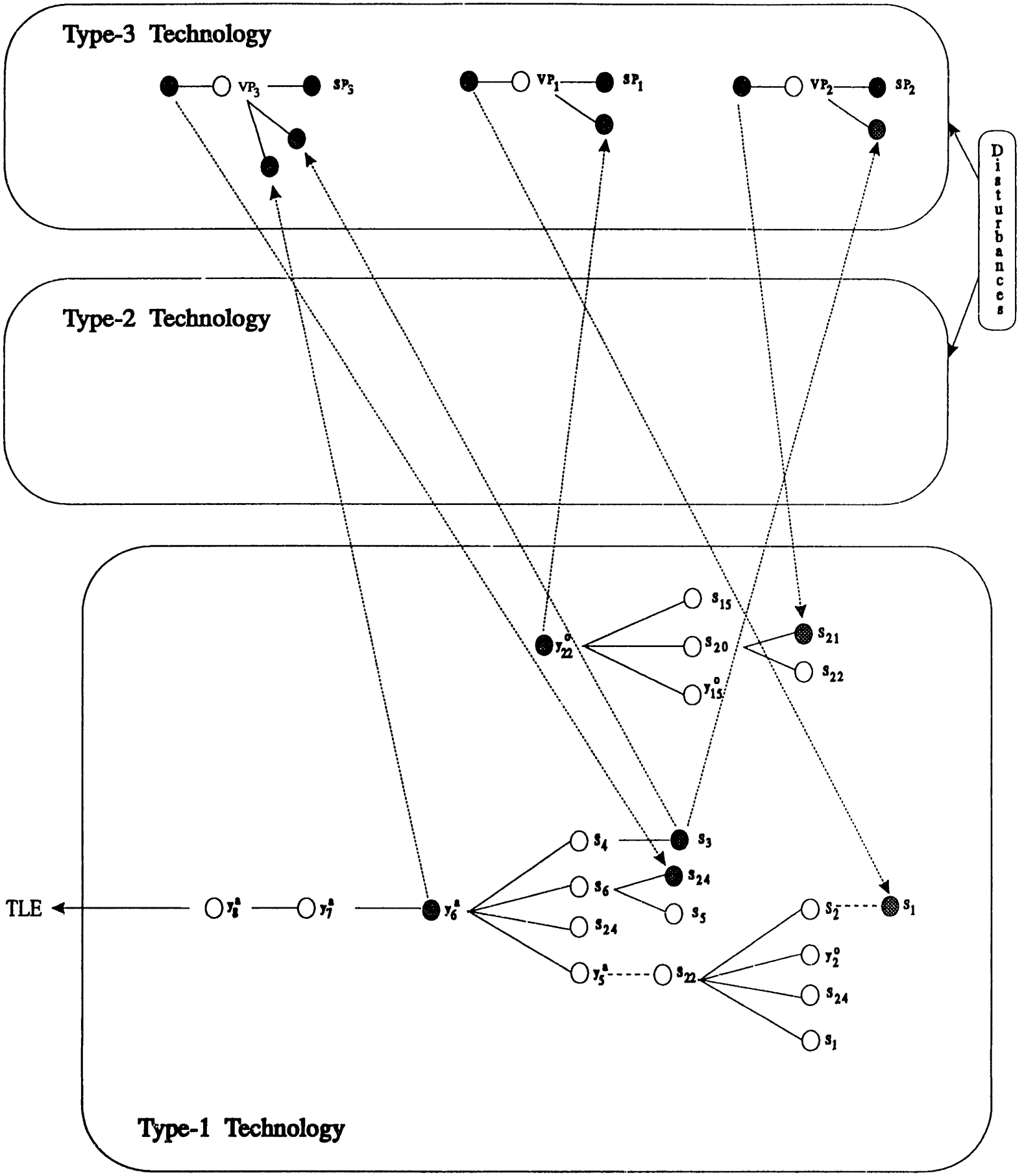


Figure 4-40: Root cause construction: y_8^a

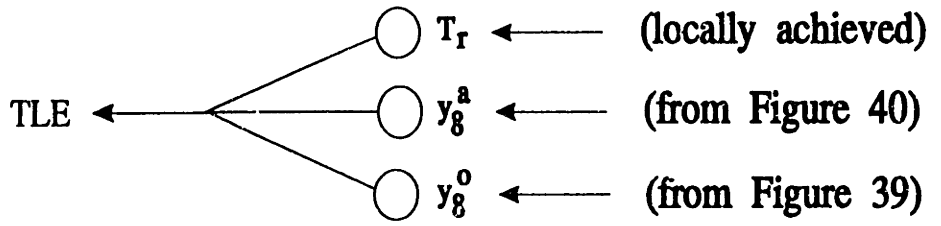


Figure 4-41: Root cause construction: TLE

TLE can be focused exclusively on the pathway leading to y^o_8. Furthermore, by reviewing control objective coverage on the nodes preceding y^a_8, demands or disturbances which can enable it are immediately identified. Variables contained in this set are: S_A, T_R, R_A, and y^a_8. Expansion of y^a_8 identifies that the control objective on y^a_6 is achieved through the manipulation of S_{24}; and, S_{24}'s constraint list identifies the associated high interlock. These prevent disturbance propagation which can enable y^o_8. However, without the interlock, the control objective on y^a_6 may not be achieved and therefore could be enabled by demands placed on S_3 or S_1, or both. Since these variables are controlled by Type-3 technologies, disturbances must enter through the inputs specified by these structures if the TLE is to be achieved.

The demand tree which results from disturbances entering through these inputs and propagating through the system is shown in Figure 4-42. It has been rearranged for illustration purposes. Three points are noteworthy concerning this illustration. Firstly, S_{21} is found in the demand tree rather than S_1. This occurs because S_3 is controlled by S_{21}, a result of Assumption-2. Secondly, as one proceeds from the root node (i.e. the TLE) to the leaves (i.e. enabling conditions) on the demand tree, information contained in the various blocks becomes more specific. Often the described demand cannot be derived from the set of process equations which describe the system (see Figure 4-28); highlighting that pathways leading to the top level event are restricted by the scope of the modeling effort (see Postulate 2). More importantly, with limited modeling efforts this will always be the case. In many processes, robust descriptions are unavailable. In this situation it is easier to associate enabling causes to those variables contained in the demand tree. This can be achieved using an external knowledge base. For example, in Figure 4-42 the demand tree extending from S_A is easier obtained from experience than it is to model. Thirdly, quantification has been used to establish demand requirements (i.e. the high or low condition).

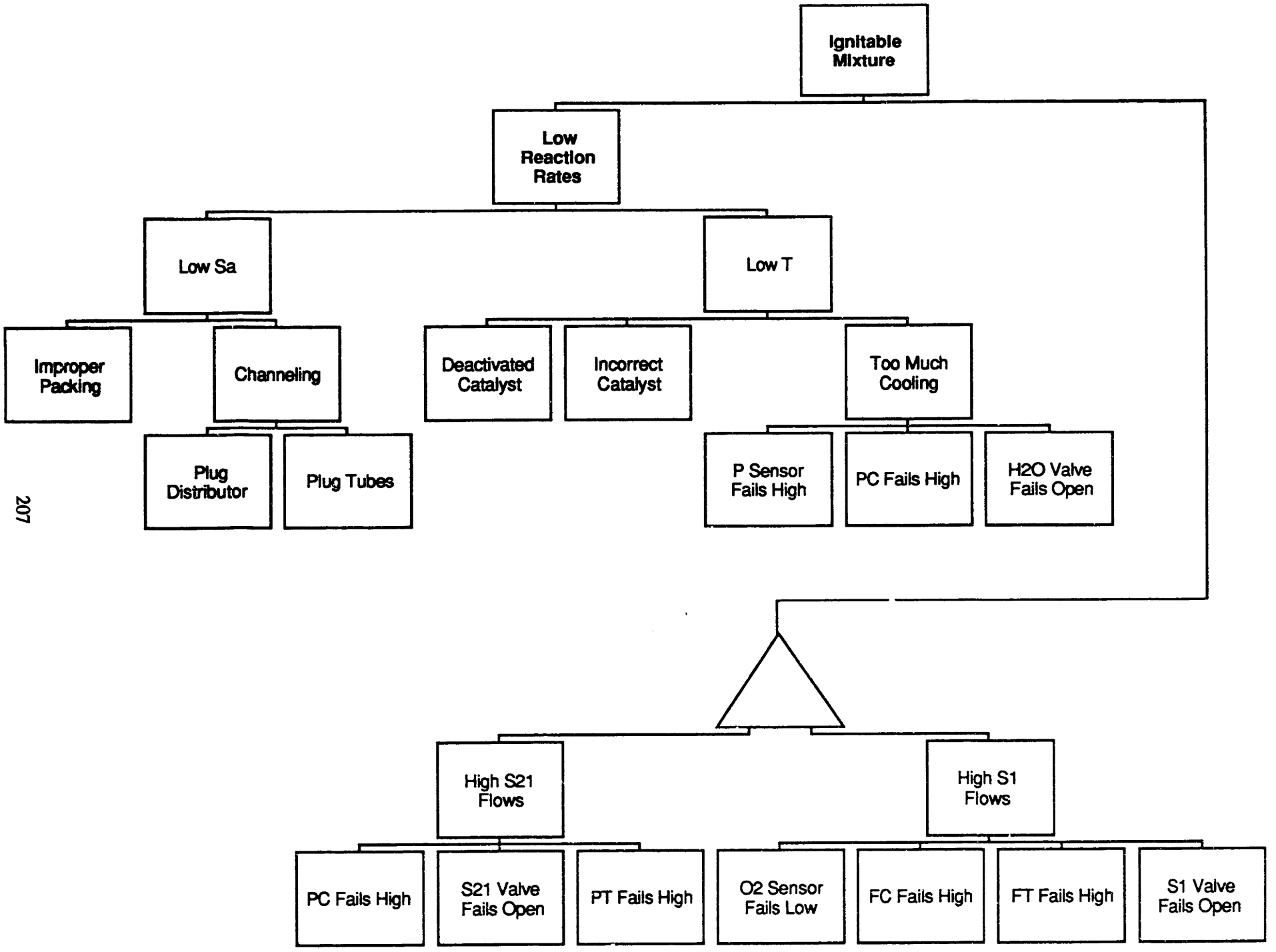


Figure 4-42: Partial demand tree for ignitable mixture in reactor

Using the algorithm presented in Section 4.2.2 a qualitative fault tree is constructed; it is a result of the demands illustrated in Figure 4-42. Quantification of the disturbances that can propagate to the top level event, and the responses required by the protective systems to allow this propagation, afford transformation of a qualitative fault tree into the partial fault tree shown in Figure 4-43. Notice that external knowledge has been used to construct this tree. This is necessary to predict demands such as: improper packing of catalyst, wrong (i.e. incorrect) catalyst, distributor plugs, and tube plugs. Furthermore, notice the association of protective systems found in the constraint list: low interlock on S_{21}. Association of such constraints with the equipment contained in the process flowsheet allow redundant control and overrides to be associated with the equipment without over specifying the system.

By associating frequency and probability data with the various demands and protective systems, a frequency for the top level event can be established. This determines the risk associated with operating the system in its current configuration. With this information a decision as to whether the system should be modified or operated as is can be made intelligently.

4.4. Conclusions

A new approach for the identification of hazards and the root causes leading to them has been presented. The approach begins with the inductive determination of hazardous reactions (see Chapter 3) and proceeds deductively through the network of processing steps to identify completely all sources of initiation within the scope of the modeling effort. Deductive identification is accomplished by constructing the topological map that satisfies the top level event in the context of the process network. By associating technology types with the variables contained in the topological map root causes and their preconditions can be identified.

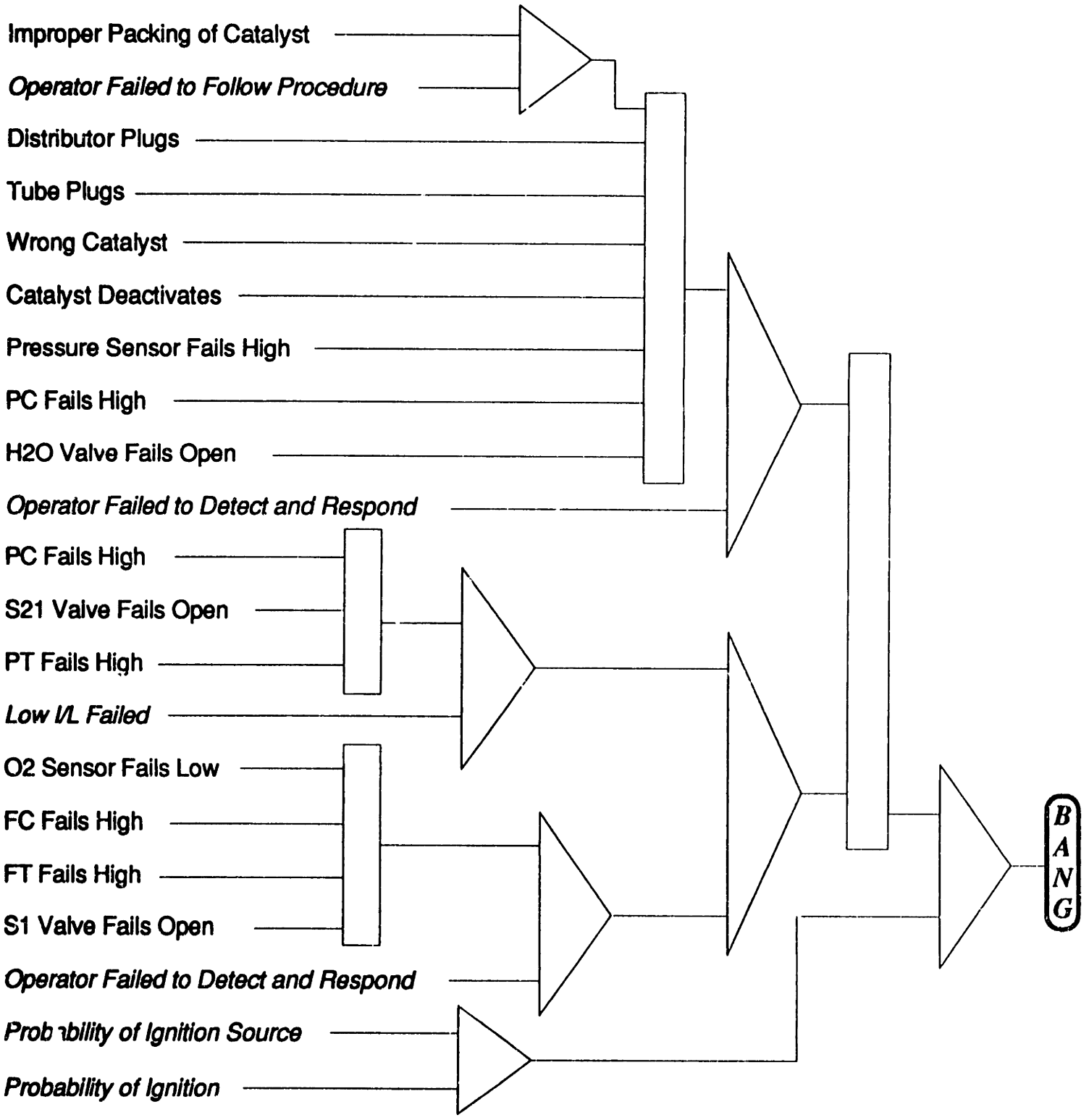


Figure 4-43: Partial fault tree for ignitable mixture in reactor

Using the knowledge contained in the underlying process modeling representation and by associating various technology types with the pathways leading to the top level event, qualitative fault trees can be constructed. An algorithm is proposed for this purpose.

The combined inductive/deductive methodological approach lifts the burden of synthetic activity from the hazard evaluation team. More importantly, it establishes a formal strategy for the integration of safety into a design technology and provides a means for discriminating among design alternatives with respect to disturbance mitigation. Furthermore, the methodology provides the basis for the optimization of a design technology with respect to the parameters which describe its inherent safety.

4.5. Bibliography

Atallah, S., "Assessing and Managing Industrial Risk", Chemical Engineering, September 8, 1980

Batstone R., International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 5.126

Battelle, Guidelines for Hazard Evaluation Procedures, AIChE Press, 1985

Boykin, R.F. and M. Kazarians, "Quantitative Risk Assessment for Chemical Operations", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.87

Carson and Mumford, "Analysis of Incidents Involving Major Hazards in the Chemical Industry", J. Haz. Mat., 1979, 3 p.149-165

Cox, R.A., "An Overview of Hazard Analysis", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, AIChE, 1987, p. 1.37

Dale, S.E., "Cost Effective Design Considerations for Safer Chemical Plants", International Symposium on Preventing Major Chemical Accidents, Washington D.C., February, 1987, AIChE, p. 3.79

Lakshmanan, R. and Geo. Stephanopoulos, "Synthesis of Operating Procedures for Complete Chemical Plants: Part I. Hierarchical, Structured Modeling for Non-Linear Planning," Computers and Chem. Engrg . 12, 985-1002 (1988).

Lakshmanan, R. and Geo. Stephanopoulos, "Synthesis of Operating Procedures for Complete Chemical Plants: Part II. A Non-Linear Planning Methodology," Computers and Chem. Engrg. , 12, p. 1003-1021 (1988).

Lees, F. P., Loss Prevention in the Process Industries, Butterworths, London, 1980

Lees, F.P., "Hazard Warning Structure: Some Implications and Applications", IChemE Symposium Series No. 80, 1983, p. J1

Lowe, D.R., and C.H. Solomon, "Hazard Identification Procedures", IChemE Symposium Series, No. 80, 1983, p. G8

Mosleh, A., Bier, V. M., and G. Apostolakis, "A Critique of Current Practice for the Use of Expert Opinions in Probablistic Risk Assessment," Reliability Engineering and System Safety, 1988, 20, p. 63

Ozog, H. and L.M. Bendixen, "Hazard Identification and Quantification", Chemical Engineering Progress, April, 1987, p. 55

Ozog, H., "Hazard Identification Analysis and Control", Chemical Engineering, February 18, 1987, p. 161

Slater, Corran, and Pitblado, eds., "Major Industrial Hazards Project Report", The Warren Centre for Advanced Engineering, University of Sidney, May, 1987

Stephanopoulos, Johnston, and Lakshmanan, "An Intelligent System for Planning Plantwide Process Control Strategies," Journal A, 29(3), p. 81-86 (1988).

Chapter 5
A Language for Reasoning about Chemical Reactivity
Part I: Formal Framework for Knowledge Representation

Summary

The analysis of traditional approaches to computer aided organic synthesis (CAOS) and chemical process development has revealed that past approaches were inherently limited by their lack of representational expressiveness. A high level computer language has been developed, based on an object-oriented environment, to address these deficiencies. It has been engineered to: (i) discover new synthetic pathways; (ii) decompose smoothly from theoretically bounded reactions to known reactions, (iii) facilitate rapid and easy extension to new reaction classes; (iv) learn to enhance search efficiency. The language is easily extended to include inorganic and organometallic chemistry. User preferences, expressed as conditionals, may be utilized at run time to focus pathway generation. The object-oriented modularity of CRL makes it easy to maintain.

The topic of computer assisted chemical manipulations is divided into two parts: Part I presents the mathematical foundations of CRL, discusses issues concerning knowledge representation, presents the types of modeling elements and the classes of modeling objects used in CRL, and discusses the semantics and the syntax of CRL. Part II focuses on accessing information contained in the modeling elements, formal construction of subclass models and composite operators, and illustrates the formal construction of reactions.

5.1. Introduction

With the exception of computational chemistry, computer-aided chemical reasoning has not enjoyed the success experienced in other scientific disciplines [1-6]. This has often been attributed to the conceptual nature of chemistry and the inexactness of known relationships [7]. However, the synthesis of new chemical reaction paths and processing alternatives that support their production remains an attractive objective within the general field of process synthesis [8]. The availability of raw materials and energy, changing ecological and health considerations, and shifting requirements of the market, reinforce each other: to create the need to identify new routes or new processes to obtain existing chemicals or develop new chemicals to meet perceived requirements.

Chemists were the first to look into computer-aided organic synthesis (CAOS) in an attempt to answer Woodwards [9,10] question and furthered by Corey [11]: "How does a chemist choose a pathway for the synthesis of a large organic molecule, given either the great diversity of organic structures and reactions, or, in contrast, the critical importance of each step to ultimate success?" The works of Corey, Wipke, Ugi, Hendrickson, Jorgensen, Moreau, Gelerner, and others are testament of the CAOS effort (see Table 1 and Fig. 5-1); excellent reviews are found in references [6, 11-40].

On the other hand, process development specialists have focused on selection of the best production route for a given chemical when alternatives exist. They are primarily concerned with the utilization of raw materials and the manufacture of relatively simple products. Their endeavors to develop new reaction paths are driven by economical, environmental, energy saving, and similar requirements. Procedures for the development of new reaction paths constrained by these requirements have been reviewed by many authors [8, 42-45].

Table 1: Computer Aids to Chemistry

Number	Authors	Prog. Name	1st Pub.
1	Corey-Wipke	OCSS	1969
2	Corey	LHASA	1971
3	Ugi-Gasteiger	--	1971
4	Hendrickson	--	1971
5	Bersohn	--	1971
6	Weise	AHMOS	1973
7	Gelernter	SYNCHEM	1973
8	Barone-Chanon	SOS	1973
9	Powers	DINASYN	1973
10	Brownscombe	EXTRUS	1973
11	Brownscombe	HEXARR	1973
12	Wipke	SECS	1974
13	Ugi-Gasteiger	CICLOPS	1974
14	Benedek	SIMUL	1974
15	Dubois	SYNOPSYS	1975
16	Whitlock	--	1976
17	Donova	HEDOS	1976
18	Powers	REACT	1977
19	Govind	REPAS	1977
20	Djerassi	REACT	1977
21	Pensak	LHASA (Dupont)	1977
22	Kaufmann	PASCOP	1978
23	Moreau	MASSO	1978
24	Gelernter	SYNCHEM 2	1978
25	Ugi-Gasteiger	EROS	1978
26	Yoneda	GRACE	1978
27	Gasteiger	PSYCHE	1979
28	Weise	GSS	1979
29	Barone-Chanon	SAS	1979
30	Stolow-Joncas	LHASA (Educ)	1980
31	Agnihotri	CHIRP	1980
32	Jorgensen	CAMEO	1980
33	Gund	SECS (Merck)	1980
34	Hendrickson	SYNGEN	1981
35	Hippe	SCANSYNTH	1981
36	Hippe	SCANPHARM	
37	Hippe	SCANMAT	
38	Zefirov	FLAMINGOES	1981
39	Ghose	--	1981
40	Schubert	ASSOR	1981

41	Zin	--	1982
42	Erdos	ASR	1983
43	Kaufmann	PSYCHO	1984
44	Seidel	--	1984
45	Hara	PFP	1984
46	Wipke	SST	1984
47	Cense	MICROSYNTHESE	1985
48	Barone-Chanon	TAMREAC	1985

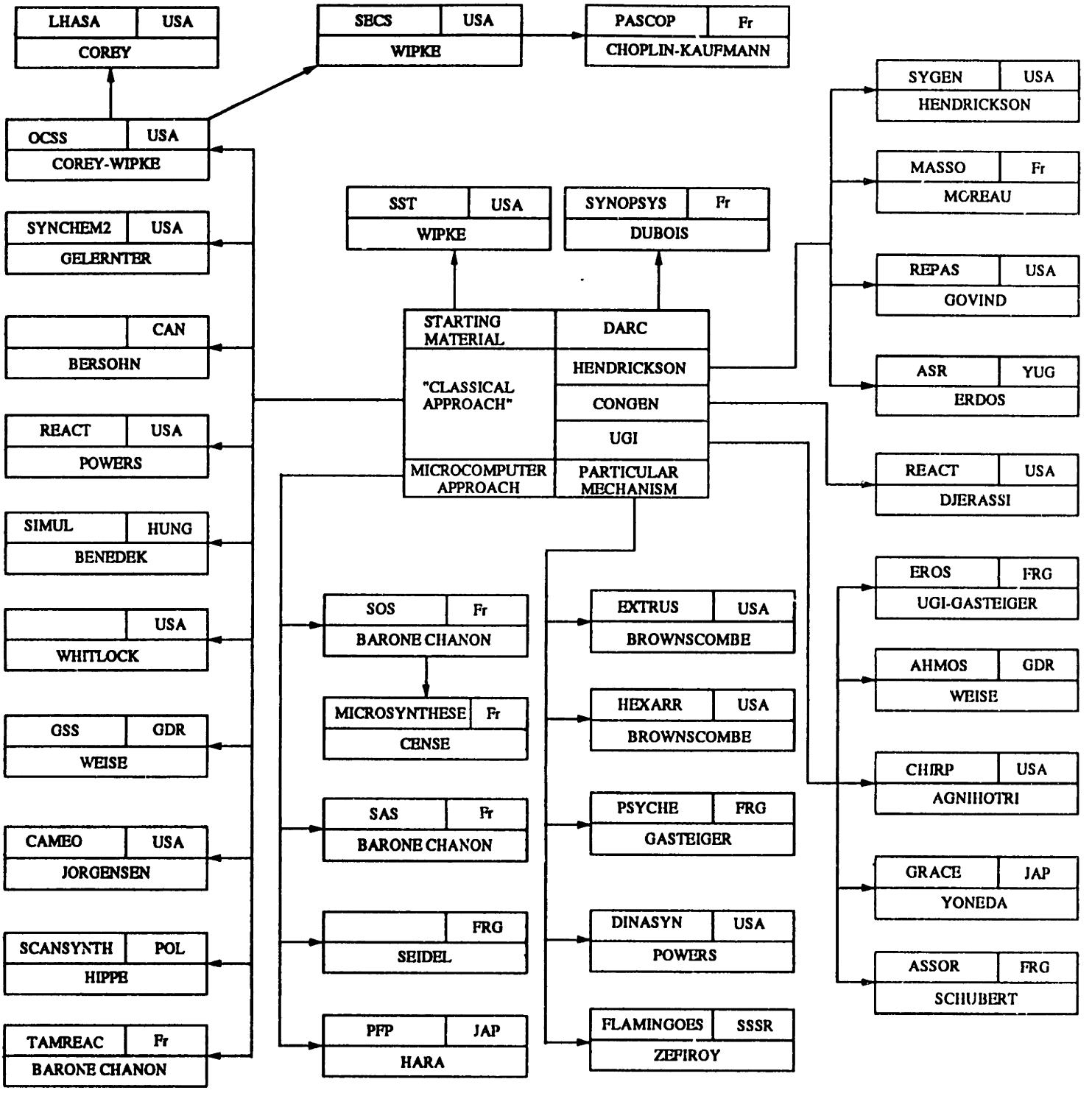


Figure 5-1: CAOS programs

But how can we generalize and group the aforementioned needs? How should chemical systems be modeled? Which questions are central to the modeling effort and how does our choice of solution affect performance and scope? Our analysis extends Wipke's classification of chemical synthesis [46] by including process synthesis objectives and suggests that all previous problems can be classified into one of the four following classes:

I	A	$\rightarrow$	?
II	?	$\rightarrow$	Target
III	A	$\rightarrow$	Target
IV	situation given	$\stackrel{?}{\rightarrow}$	situation desired

For each of these classes the criteria may vary, the quality and quantity of required data will vary, synthetic strategies will be different but the issues - the questions central to the advancement of computer-aided chemical reasoning - will remain the same: how should molecules and reactions be represented, what degree of completeness is necessary and what is the cost for achieving that degree of completeness, what synthesis strategies should be employed, what criteria for evaluation should be used, how is the analysis to be conducted, and what is the impact of computer hardware?

Since the early efforts, computer-aided modeling has been generally organized around unilateral computations that perform predetermined operations on fixed inputs to yield desired outputs. This approach has highlighted a series of inherent disadvantages: The time and cost associated with the computer model development are high; The resulting models are difficult to document and maintain accurately; The reuse of computer-aided models is minimal: they tend to be task-specific and are often intrinsically linked to the solution procedures; The models cannot be synthesized automatically by the computer in

the course of automatic execution of a task; Interactive modeling requires a highly skilled programmer. These weaknesses result in an enormous duplication of effort.

Accumulated modeling knowledge is almost impossible to use since the underlying modeling context varies with purpose, assumptions, and simplifications; these are rarely documented and rationalized. Moreover, the fragmentation of modeling efforts and the ad hoc character of their computer implementation has led to internal inconsistencies among the models used in different chemical tasks. The sheer diversity of models used to support synthesis related tasks [6] exemplify this phenomena. So why must every modeling effort start from scratch [47]?

5.1.1 Premise of traditional chemical reasoning systems

All of the above weaknesses can be contributed to a *lack of representational expressiveness*. The fundamental premise of traditional computer-aided chemical reasoning systems is that unsuccessful efforts have been the result of delinquent paradigms. We have found the contrary to be true: representational limitations of the modeling effort have prevented the richness of the chemical domain from being captured and fully exploited. For example, past models were static and often one-dimensional in their representation of knowledge (e.g.tables, records, matrices, procedural models). Often they did not or could not *explicitly* include information such as the following: (i) Underlying assumptions on reaction site selection and electronic state.(e.g. resonance form). (ii) Simplifications made by the modeler to limit the model's validity over a given range of conditions or to underscore the relative importance of various physicochemical phenomena. (iii) Missing relationships including qualitative relationships, order-of-magnitude reasoning, and inequalities. These are not normally part of the chemical models because conventional analytical techniques cannot handle them. (iv) Scope of synthetic task, that is, what was the model intended for.

These technological restraints and limits prevented declarations and procedural information from being captured and reasoned about efficiently. Often the declarative knowledge which is articulated in a model is intrinsically integrated with the procedural knowledge (i.e. the "how to ..." methodologies); and, this knowledge depends on the specific characteristics of the modeling relationships. Consequently, if the modeling relationships change the declarative knowledge must be disentangled and altered to support the model. An evolutionary modeling effort requires, therefore, that declarative knowledge be completely decoupled from procedural knowledge.

Such inherent deficiencies instilled rigidity into the developed systems. Not only were representations limited in the type of knowledge that they could capture but the resulting structures prevented modularity and flexibility; they were incapable of evolutionary design. Efforts in engineering science reinforce these observations [48]. Many tasks are not achievable if the representation isn't sufficiently rich to express the necessary concepts [49]. Concepts must be manipulated directly if powerful reasoning is to be achieved. The success of any advanced computer-aided tool for enhancing the reasoning and manipulation of chemical structures requires: (a) the development of a representation sufficiently rich to embody advanced scientific concepts, and (b) a means for manipulating these concepts and reasoning about them directly.

5.1.2 Outline of the Chapter

The framework used to formalize our concept of chemical operations and the language used to manipulate chemical structures is divided into two parts. Part I presents the formal framework for knowledge representation, discusses the types and classes of modeling elements, and presents the semantics and syntax of the language. Part II focuses on the usage of the language: describing how information if accessed, illustrates formal construction of subclass models, and demonstrates reaction construction.

Part I, the focus of this chapter, has been organized as follows: Section 5.2 presents an overview of the language; Section 5.3 describes the modeling of chemical knowledge; Section 5.4 provides a summary of the language's characteristics.

5.2. Languages and the Modeling of Chemical Systems

Just as Lavoisier stated two hundred years ago: "*It is time to rid chemistry of obstacles of every kind this reform must be brought about by perfecting the language*;" we postulate that a high level (computer) language must be developed for meaningful advancement to occur in computer aided chemical reasoning. Indeed, by defining appropriate primitives, means of combination, and means of abstraction, engineers have created specialized problem-oriented languages which provide improved modeling facilities for the particular problem at hand [50]. For example, the language of "electrical networks" emphasizes the physical modeling of devices in terms of discrete electrical elements (e.g. resistors, capacitors). In contrast, the primitive objects of "electrical systems" are signal processing modules (e.g. filters, amplifiers), underlining the functional behavior of the resulting models. Similarly, in civil engineering we find languages [51-52] which allow the declarative modeling of structures (made up of gilders, rods, plates, beams, shells) and their properties (e.g. stress or strain in structural elements). In chemical process engineering we see an evolution in modeling from the language of "unit operations" to the language of "elementary physical and chemical phenomena". This is due to the inadequacy of unit operations to formalize a large and growing number of specialized processing units.

Domain-specific modeling languages are "very high level" and of special purpose. They have a *theme*, that is, the class of ideas which is optimized to communicate [53]. They allow the user to employ terms and constructs which lie closer to the informal terminology and modes of speech customary in the discussion of domain-specific problems; but, in constructing such a language one should strive for domain-specific

generality of the language. Thus, the same modelling language should satisfy all the needs: chemical synthesis, reaction analysis, pathway optimization, and innovative design of reaction or processing networks. The implication of this requirement is clear, "*a modeling language in chemical reasoning should be fully declarative and in no way should its generality be compromised by the specificity of the methodologies of the chemical tasks themselves*".

The language described herein, for chemical reasoning and manipulation, satisfies these requirements. The specialized language developed for reasoning about associated declarative structures is not an ad hoc construction but is based on a clear formalism and satisfies the basic premises of grammar, vocabulary and semantics upon which programming languages are founded.

5.3. Overview of the Proposed Modeling System

In the previous sections, we highlighted three essential criteria that advanced modeling systems should satisfy, namely: (1) all declarative models should be fully articulated, (2) declarative knowledge should be completely decoupled from procedural knowledge, (3) a modeling language should be provided to allow the user to think about the task as hand in terms that are familiar. For a computer-aided chemical reasoning system we add a fourth (4) criterion: the logical structure of chemistry should be exploited by imbedding natural constraints into the declarative model. In addition, the language proposed in this chapter has been designed to achieve the following major objectives:

(a) Discovery. The system is capable of identifying theoretically feasible reactions and synthetic routes. The generation of reactions and the ensuing pathways can be restricted through user specified constraints (e.g. $\Delta G \geq 10$ kcal/mol or toxicity less than α as calculated by EP toxicity test).

(b) Rapid and easy extension to new reaction classes.

(c) Smooth decomposition from theoretically bounded reactions to known reactions. The system provides both an upper bound on the set of potential pathways optionally restricted through a set of user specified constraints and a lower bound specifying known transformations.

(d) Learning to enhance search efficiency. Abstract reactions can be constructed from reaction segments as can abstract pathways from pathway segments. These new sequences are archived and used directly by the system in future searches.

(e) Represent a chemical pathway at any level of detail by using multiple coexisting levels of abstraction which can communicate between each other and which can explicitly keep track of interrelated units, and their associated modeling relations. The ability to represent multiple expressions of the same chemical species is a critical feature of the system that distinguishes it from other systems.

(f) Automatically generate the set of basic relationships (species, balances, reaction and transport conditions, equilibrium equations, kinetics, etc.) that describe the system. This requirement implies the development of explicitly structured mathematical models, involving variables, terms and relationship objects.

(g) Capture and utilize qualitative, semi-quantitative relationships (ordinal, order-of-magnitude), or boolean relationships. Such requirement will allow the modeling system to be used beyond the scope of traditional systems.

(h) Offer explicit documentation of all the hypotheses, assumptions and simplifications that give rise to a particular model, pathway, or reaction. The model should have all the characteristics of the knowledge it embodies - it should be transmittable from one person to another and open to modification, improvement and combination with other knowledge. Thus it should allow a direct mapping between the ideas of the chemical model and the knowledge-base of a project.

The above requirements go far beyond the scope of traditional computer-aided modeling systems for synthetic purposes. They require that the models produced be explicit, transparent and ready to be used by the chemist without going to literature to check for assumptions, input-output language conventions, and without performing obscure tricks to cope with rigidities.

We propose nine modeling elements for capturing knowledge requisite for chemical reasoning. These elements provide the vehicle for decoupling procedural knowledge from declarative knowledge. Communication between the modeling elements is accomplished through a rich set of semantic relationships that adhere to strict grammatical rules. A class hierarchy is presented for efficiently managing tasks, concepts, and exceptions or rules associated with the modeling elements.

The framework used to formalize our concept of chemical operations is presented in the next section. Therein we use lower case to denote data and upper case to denote operations. Objects[1] are represented in bold; class specific functions (i.e. procedures that

[1]An object is an instance of an abstract data type. Each data type has a set of operations closely associated with it that define the interaction of the data elements representing the structure to the external world. An object is given meaning in an environment through the attributes which describe it. Methods associated with an object allow its attributes and the value of those attributes to be dynamically modified, added, or deleted.

evaluate only on members of the same class) are built as *methods* on an object class. This architecture limits domain complexities.

5.4. Modeling Elements for Defining Chemical Structures

The efficient representation of chemical species and the reactions that they undergo is paramount to any chemical reasoning program; but how can the richness of the chemical domain be expressed?; how should the knowledge of chemistry be incorporated into such a scheme and how does one identify the correct data structures or the correct modeling elements? Most importantly, how might these modeling elements be used to construct a rich vocabulary of chemical terms that have meaning?

Such begins by first *decoupling* the classification of a chemical structure from its observed reactivity (Fig. 5-2). This allows multiple (chemical) behaviors to be associated with a single chemical structure[2]. Since the observed behavior exhibited by a structure is context dependent (e.g. depending on the reaction conditions), the selection of a characteristic behavior may be deferred until the reaction environment is established. Reaction alternatives can then be pruned *before* generation. Second, basic transformations called composite operations, are broken down into its functional components (Fig. 5-3). These components are then analyzed and processed to create a reaction.

We have chosen a graph theoretic representation of chemical structures wherein nodes are atoms and edges are bonds; nodes and edges are represented as objects. The description of a modeling element or chemical structure varies depending on the level of specification. Three modeling elements are used to create chemical structures; they are given below. Each modeling element is described by a set of attributes and a set of procedures called methods that operate on the model class.

---

[2]For example, an alcohol can act as a bulk solvent, weak acid, or weak nucleophile.

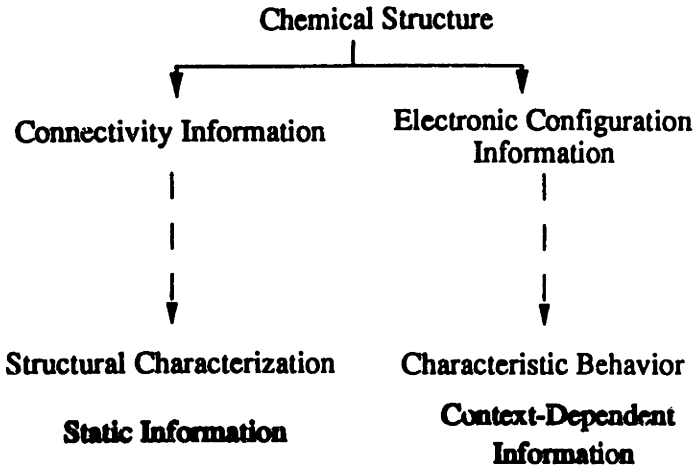


Figure 5-2: Representation of chemical knowledge

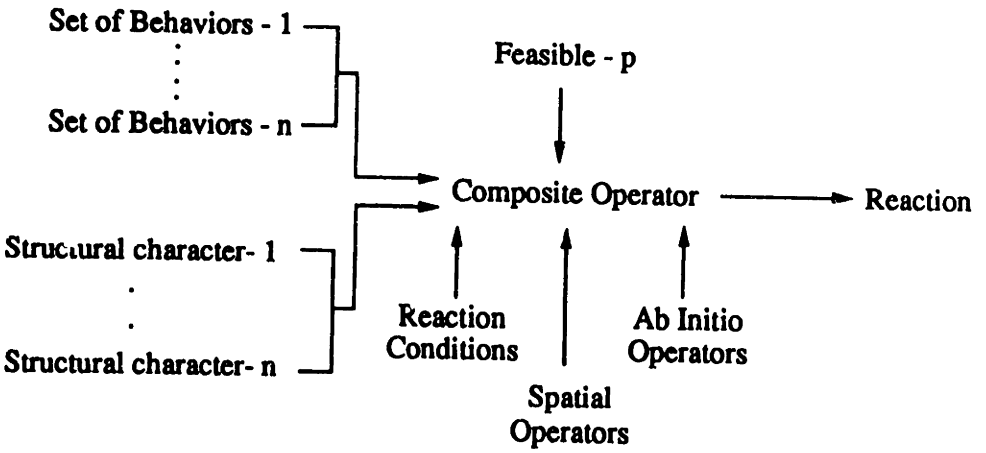


Figure 5-3: Schema for reaction generation

Modeling element 1: atom. An atom is the building block of chemical species; an atom is any entity found in the periodic table. Each atom encapsulates its own structural information, modeling relationships, and modeling assumptions. An atom knows its neighbor atoms and the bonds that constitute its connection to those neighbors.

The attributes describing the object class atom are shown in Figure 5-4; additional attributes may be associated with an object's class to refine the class description. For example, the attributes: formal charge, oxidation state, steric hindrance, chirality, etc. may be included into an atom's description to refine the class description. Several of the attributes given in Figure 5-4 warrant further explanation: attributes identifier and old-identifier, are pointers used to trace an atom's lineage; attribute database atom contains an object **db-atom** that manages invariant information (Fig. 5-5) indigenous to a particular atomic species (e.g. atomic-symbol, atomic-number, atomic-weight, etc.). The attribute parent-abc associates an atom to a given structure; its value provides the vehicle for moving between the atom representation and the parent-abc representation. Parent-group is used similarly, an **atom** may have an association with a group or set of groups as well as a parent-abc. As in the case of parent-abc, the value of parent-group provides a conduit for accessing information residing at the **group** representational level. As will be shown later, all representational levels can be accessed from any individual level. The freedom of movement between levels enhances representational expressiveness and facilitates efficient reasoning.

A partial listing of the methods operating on **atom** is shown in Figure 5-4. Methods with the "compute" prefix (e.g. compute-parent-groups, compute-hybridization, and compute-formal-charge) are used to evaluate attribute values. Predicate methods (i.e. methods returning a boolean value) are represented by the "-p" suffix. These methods are helpful when a characteristic of the atom is used by several different procedures, whether they be

atom attributes

- identifier
- old-identifier
- name
- type
- chiral-p
- formal-charge
- electronegativity
- electron-withdrawing-substituent-p
- acidity
- connectivity
- hybridization
- conjugated-p
- p-orbitals
- open-approach-p
- radical-orbital
- bond-angle
- parent-abc
- progenitors
- parent-groups
- neighbor-atoms
- neighbor-groups
- bonds
- db-atom

atom methods

- compute-connectivity-number
- compute-parent-groups
- compute-neighbor-groups
- compute-hybridization
- compute-p-orbitals
- compute-formal-charge
- compute-radical-orbital
- identify-electron-withdrawing-substituent
- find-atom-chains
- atom-backbone
- atom-degree
- higher-degree-atom-p
- compute-open-approach-p
- identify-conjugated-p
- atom-specific-selections
- create-atom
- abstract-grouping
- 1,2-mobile-atom-p
- 1,5-mobile-atom-p
- mobile-univalent-atom-p

Figure 5-4: Select attributes and methods of atom

db-atom attributes

- name
- atomic-symbol
- atomic-number
- atomic-weight
- valence
- row
- column
- orbitals
- valence-electrons
- electronegativity

Figure 5-5: Select attributes of data base atom

internal or external to the object class which contains them. For example, various operations may require knowledge about an atom's mobility as in the case of radical rearrangements; rearrangement alternatives are accessed using methods illustrated by 1,2-mobile-atom-p and 1,5-mobile-atom-p.

The remaining methods evaluate properties of an atom or properties of the parent structure. For example, atom-backbone identifies the skeleton of interest; find-atom-chains enumerates all the paths emanating from the specified atom; abstract-atom-grouping produces metagroups around reaction centers: partitioning of the parent structure into active and inactive sites enables efficient manipulation during pathway construction. New atoms are constructed using create-atom. Selector functions, implemented as methods, are also built into **atom**. These methods provide an efficient means of collecting **atoms** which contain some specified desired property. Illustrative examples of selector methods include: collect-sp2-atoms, collect-oxygen-atoms, collect-beta-neighbors, collect-terminal-atoms, etc.

> **Modeling element 2: bond.** Atoms are connected by bonds. Like atoms, each bond has associated with it structural information and models that manipulate this information to deduce new features which describe it. Bonds know the atoms that describe it. Information is transferred from entity to entity, and new connectivity information is deduced through these elements (bonds).

Bond attributes are similar in spirit to those given in Figure 5-4. A partial listing of attributes describing and methods operating on object class **bond** are shown in Figure 5-6. Notice that the atoms attribute value allows movement from the bond representation to an atom representation while the parent-abc attribute value allows movement between various levels of abstraction within the representation (e.g. groups and structures). In

bond attributes

- identifier
- character
- strength
- length
- type
- atoms
- parent-abc
- progenitors
- db-bond

bond methods

- methods
- find-bond-chains
- cleave-bond
- compute-bond-strength
- remove-bond
- add-bond
- modify-bond
- create-bond

bond-selectors

- same-bond-type-p
- bond-equivalence-p
- alpha-bonds
- beta-bonds
- gamma-bonds
- equivalent-bonds-p
- terminal-bond-p
- internal-neighbor-bonds
- terminal-bond-for-additions-p

Figure 5-6: Select attributes, methods and selectors of bond

addition to evaluating attribute values, bond methods are used to facilitate the construction of chemical structures. To achieve this purpose cleave-bond, add-bond, remove-bond, modify-bond, and create-bond are provided. Like atom selectors, bond selectors are used to collect **bonds** exhibiting a designated property to facilitate processing.

> **Modeling element 3: atom-bond-configuration.** Atoms and bonds containing connectivity information and spatial relations comprise an atom-bond-configuration. All chemical structures can be defined by an atom-bond-configuration. This structure, although in the strictest sense is not a primitive, has been elevated to a primitive to reduce complexity in subsequent reasoning. An atom-bond-configuration has associated all methods and properties that are indigenous to an abstract chemical structure, such as physical and spatial properties. Thus, each specialized chemical structure is constructed from an atom-bond-configuration. An atom-bond-configuration in conjunction with the comprising atoms and bonds allow us to capture and isolate the structure features of a configuration from the features that characterize its behavior.

Generic chemical structures are represented by the object class atom-bond-configuration. The attributes describing this class include name, identifier, atoms, bonds, empirical formula, frontier molecular orbital (FMO), etc. Since atoms and bonds are objects, the value of the attributes describing these entities are the set of objects comprising the atom list and the bond list respectively. As a consequence, the values of the attributes describing these objects are easily accessible to procedures invoked by the atom-bond-configuration. For example, the evaluation of an atom-bond-configuration's highest occupied molecular orbital (HOMO) requires evaluation of the atomic orbitals (AO's) which comprise it. This information, being resident in the atoms comprising the atom-

bond-configuration, is accessed through selector functions which are applied by the compute-HOMO procedure of atom-bond-configuration.

The value of additional attributes may advantageously be represented by objects as well. For example, we may wish to know the progenitor(s) of a particular structure. The progenitor, being an instance of atom-bond-configuration, may be placed directly into the value holding slot of attribute "progenitor". By accessing the values of the attributes describing the progenitor (i.e. atom-bond-configuration attributes and attributes of other specializations such as organic-molecule), a complete account of its trajectory can be made available.

Attributes common to an **atom-bond-configuration** are shown in Figure 5-7. Select methods operating on an **atom-bond-configuration** are shown in Figure 5-8. Methods of particular interest include general setup methods, map-old-abc-to-new-abc, equivalent-atom-bond-configuration-p, and identify-atom-bond-configuration-environment. Setup methods, as expected, provide a means for instantiation as discussed above. Map-old-abc-to-new-abc is a utility method which maintains the system pointers; an important feature when competing pathways are simultaneously analyzed[3]. Equivalent-atom-bond-configuration-p determines when two abc's are equivalent while identify-atom-bond-configuration-environment accesses the reaction environment of an **abc**. Although the reaction environment is specified for most pathways of synthetic interest, in the broader domain of computer-aided chemical reasoning many systems exist in which the environment is not known a priori. Hazard identification is an obvious example.

[3]When multiple competing reaction systems coexist, this method is particularly useful in aiding the assessment of hazard forming pathways.

atom-bond-configuration attributes

- identifier
- atoms
- bonds
- empirical-formula
- molecular-weight
- charge
- equivalent-atoms
- equivalent-bonds
- weakest-bond
- heat -of-formation
- entropy-of -formation
- free-energy-of-formation
- homo
- lumo
- progenitors
- environment
- ...

Figure 5-7: Select attributes of atom-bond-configuration

atom-bond-configuration-methods

setup-atom-bond-graph-descriptor
atom-bond-graph-descriptor
setup-atom-bond-graph-from-old-graph
setup-atom-bond-configuration
make-graph-from-symmetric-connectivity-matrix
make-bonds-from-connectivity-list
make-atom-bond-configuration
equivalent-atom-bond-configuration-p
map-old-abc-to-new-abc
identify-bond-printed-representation
correct-bond-number-p
atom-symmetry-identification
bond-symmetry-identification
enumerate-all-atom-chains
identify-atom-bond-configuration-environment
identify-atom-bond-configuration
compute-equivalent-bonds
compute-weakest-bond
compute-equivalent-atoms
...

Figure 5-8: Select methods operating on atom-bond-configuration

5.4.1 chemical structure instantiation

The modeling language developed provides two vehicles for creating **atom-bond-configuration's**. Chemical structures can be generated from either be-matrices or atom and bond tables. The information contained in these representational structures is parsed to create **atoms** and **bonds**. Each object contains the primitive data supplied by the representation table and methods used to derive additional attributes/features. The **atoms** and **bonds** comprising the **abc** graph contains the information and procedures required to reason about them directly. A particular spatial orientation, for example, is achieved by modifying the values of these attributes; algorithms requisite to this task are specified elsewhere [54-56]. Once an initial set of structures is provided new **abc's** are created directly from the **atoms** and **bonds** participating in the reaction. Composite operators perform the necessary bookkeeping for molecular generation.

A chemical structure or **atom-bond-configuration**, **abc**, is derived from **atoms** (a_i), **bonds** (b_i), and spatial connectivity information (c_i). The structural representation provided is parsed using a set of ABC operations. These operations are represented as:

$$\text{atom-table + bond-table} \xrightarrow{ABC} \mathbf{abc}$$

$$\text{be-matrix} \xrightarrow{ABC} \mathbf{abc}$$

$$\mathbf{atoms + bonds} \xrightarrow{ABC} \mathbf{abc}$$

or:

$$\mathbf{abc}_0 = ABC(a_0, b_0, c_0)$$

...

$$\mathbf{abc}_i = ABC(a_i, b_i, c_0)$$

All procedures that are generic to abc are contained in ABC. Examples of methods and the attributes/properties used to describe them may include:

1. structurally derived information (spatial orientation, steric effects, connectivity, molecular weight, etc.)

2. molecular orbital information (HOMO, LUMO, etc.)

3. physical property information (ΔH_f, ΔG_f, T_m, T_b, etc.)

An important function of the ABC is to manage where **atoms**, **bonds**, and **abc's** originate so that pathways can be easily constructed.

5.5. Modeling Elements for Defining Chemical Transformations

The functional components shown in Figure 5-3 lie at the heart of every chemical transformation; several are modeling elements. The set of operations defining a reaction access the information contained in these modeling elements. The scope and extent of the reaction space generated is defined by the knowledge encoded into the modeling elements.

> **Modeling element 4: chemical-behavior.** The electronic state that characterizes chemical reactivity is captured in chemical behavior. This may result from internal or external influences or both. Assessment of these electronic states characterize radical, nucleophile, and electrophilic behavior or combinations thereof.

The set of behaviors that defines the spectrum of a species reactivity is established by evaluating the electronic state with respect to internal and external influences. These influences may be established by the ground state, the excited state, or the effect of an

external environment on the state. The framework used to establish chemical behavior is given below. Information flow is given in Figure 5-9.

$$\mathbf{acb} \xrightarrow{S} \mathbf{s}$$

$$\mathbf{s} \xrightarrow{I} \mathbf{i}$$

$$\mathbf{i}_1 \ldots \mathbf{i}_k, \mathbf{s}_1 \ldots \mathbf{s}_k \xrightarrow{E} \mathbf{e}$$

$$\mathbf{i, e, s} \xrightarrow{CB} \mathbf{cb}$$

Here S is the set of operations used to deduce and assess structural character, s, from an abc; I is the set of operations used to evaluate intrinsic electronic character, i, from s; E is the set of operations used to assess the electronic character of the external environment, e, from $i_1 \ldots i_k$, $s_1 \ldots s_k$, where k is the number of species present; CB is the set of operations used to evaluate i, e, s to establish the set of potential chemical behaviors, cb. These operations comprise the basis for reactivity assignment and are distributed within the model element cb. They are used to classify electronic states (see Section 5.6) and derive properties associated with an electronic state: radical behavior is a derived property of singly occupied molecular orbital.

An important feature of cb is that detail can be managed and pursued on demand to assist in an evaluation. The structure permits the association of a set of potential behaviors with a species. These tasks can be performed at run time and the assignment made *dynamically*. For example, whether an alcohol acts as a bulk solvent, weak acid, weak nucleophile, or form an alcoholate anion, is dynamically assigned by chemical-behavior (cb) once the reaction environment is known.

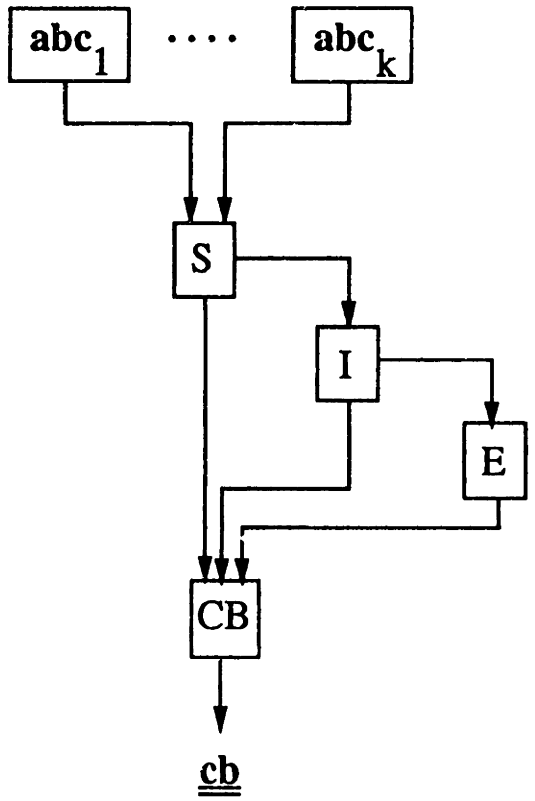


Figure 5-9: Information flow

Modeling element 5: reaction-environment. The set of properties characterizing the environment in which a reaction is occurring is contained in reaction-environment.

The physicochemical properties used to describe a reaction system and contained in **reaction-environment**. Attributes typical of this element include temperature, pressure, wavelength, surface type, species concentration, pH, species present, etc.; methods built into the modeling element evaluate attribute values and access information in other modeling elements.

Modeling element 6: ab-initio-operator. Low lying transformations which are characterized by bond cleavage, bond formation, and electron distribution are captured in ab initio operator. These transformation may be grouped as mass transfer operations and energy transfer operations for abstraction purposes. Axioms, be they laws, rules, or constraints, obtained from the physical sciences are encoded to prevent the generation of an atom-bond-configuration obtained by applying infeasible transformations. Such knowledge may include conservation of mass, conservation of energy, charge support, Pauli exclusion principle, valence constraints, etc.

A specific chemical transformation or rearrangement is accomplished by calling upon a set of base operations, called ab initio operators (K_{ai}'s), to perform the necessary elementary structural changes and electron redistribution. The base operations selected are evaluated on the reaction sites of the specified reactant(s). These low lying operations have imbedded models or constraints, such as conservation of mass and energy, that prevent the generation of infeasible structures as specified by the scope of

knowledge which defines them[4]. We have found that the knowledge embedded in these models is nearly always structural in nature reflecting for example, knowledge about orbital symmetry, atom coordination, or charge. As will be discussed later, $\mathbf{K_{ai}}$'s can be used directly to generate the upper bound of theoretically feasible transformations.

> **Modeling element 7: composite-operator.** A composite-operator contains knowledge about user specifications and mechanistic operations. Generic rate information (i.e. relative magnitude of rate constants; for example, rate constants of photochemical reactions lie between 10^{-10} and 10^{-12} sec) and electronic state information (e.g. excited, radical, charged) are also contained in **composite-operator**. This information is used to limit an operator's range of applicability.

A composite operator is a procedure which transforms a set of chemical species of predisposed behaviors into a set of products, provided an optional set of prespecified conditions are satisfied. These conditions can be specified by the user or imposed by the system. They may encompass virtually any symbolically encodeable concept: structural character, chemical behavior, spatial orientation, reaction conditions, free energy requirements, enthalpy considerations, toxicity, etc. A composite operator is composed of a feasibility predicate, site selection, and a method for transforming these sites (via successive application of ab initio operators). The procedures that comprise it are defined below. Notice that the architecture completely decouples procedural knowledge from declarative knowledge.

definition: A composite operator, K, is composed of K_f, $K_{get\text{-}sites}$, and K_t. K identifies potential reaction sites by calling

[4]For example, if we have not specified operators capable of constructing delocalized bonds then we should not expect structures which emanate from this knowledge.

$K_{get\text{-}sites}$ on the structure(s). K_t is the applied to those sites forming a new set of **abc's** when K_f returns true.

definition: K_f is the feasibility predicate for **K**; it returns a boolean value of true when the encoded set of prespecified conditions are achieved.

definition: K_t is a procedure that utilizes one or more K_{ai}'s to transform a set of reaction sites. Successful application of the K_{ai}'s generate a new {**abc**} denoted {**abc***}. Physical laws encoded in K_{ai} help to restrict the number of elements in {**abc***}.

definition: $K_{get\text{-}sites}$ identifies potential sites by calling the **cb** of each **abc** to access chemical reactivity.

Mappings established by these operations are:

$$\mathbf{abc}_1 \ldots \mathbf{abc}_k \xrightarrow{K_{get\text{-}sites}} \text{sites}$$

$$\{\mathbf{abc}\},\{\text{sites}\} \xrightarrow{K_t} \{\mathbf{abc}^*\}$$

$$(\mathbf{abc}_{in,1} \ldots \mathbf{abc}_{in,k}) \xrightarrow{K} (\mathbf{abc}_{out,1} \ldots \mathbf{abc}_{out,m})$$

The feasibility predicate, K_f, is described using:

$$K_f: \quad \left(\prod_{i=0}^{k} I(\mathbf{abc}_i), \prod_{i=0}^{l} E(\mathbf{abc}_i), \prod_{i=0}^{k} S(\mathbf{abc}_i), \text{operating-conditions}\right)$$

$$\Rightarrow \text{boolean}$$

that is,

K_f: **(cb, operating-conditions)**

$\Rightarrow$ **Yes/No**

Although we have used **K** to represent known transformations, new composite operations can be built directly (i.e. without specifying K_t) upon $\mathbf{K_{ai}}$. These new operators, denoted as $\mathbf{K^*}$, embed user requirements in the procedure feasible-p. Unlike **K**, $\mathbf{K^*}$ is capable of discovering new pathways and products subject specifications supplied by the user. Figure 5-10 illustrates this point: $\mathbf{K_{theor}}$ defines the theoretical reaction space that results from mathematical combinations; $\mathbf{K_{ai}}$ tightens the $\mathbf{K_{theor}}$ bound by adding restraints (e.g. physical laws); $\mathbf{K^*}$ tightens the $\mathbf{K_{ai}}$ bound by adding user requirements; **K** represents known transformations and name reactions.

5.6. Modeling Elements for Defining Reactions and Pathways

The successful application of a composite operator on a structure or group of structures is a reaction; or, more formally:

definition: A reaction (R) under K is the successful mapping from an abc to a new abc. For example, R_1 represents the mapping of abc to abc_1 by K, as shown below:

$$\{abc\} \xrightarrow{K} \{abc_1\}$$

Modeling element 8: reaction. Transformation specific information is captured in modeling element **reaction**. A **reaction** knows which species are reactants and which species are products. It knows which **K** is

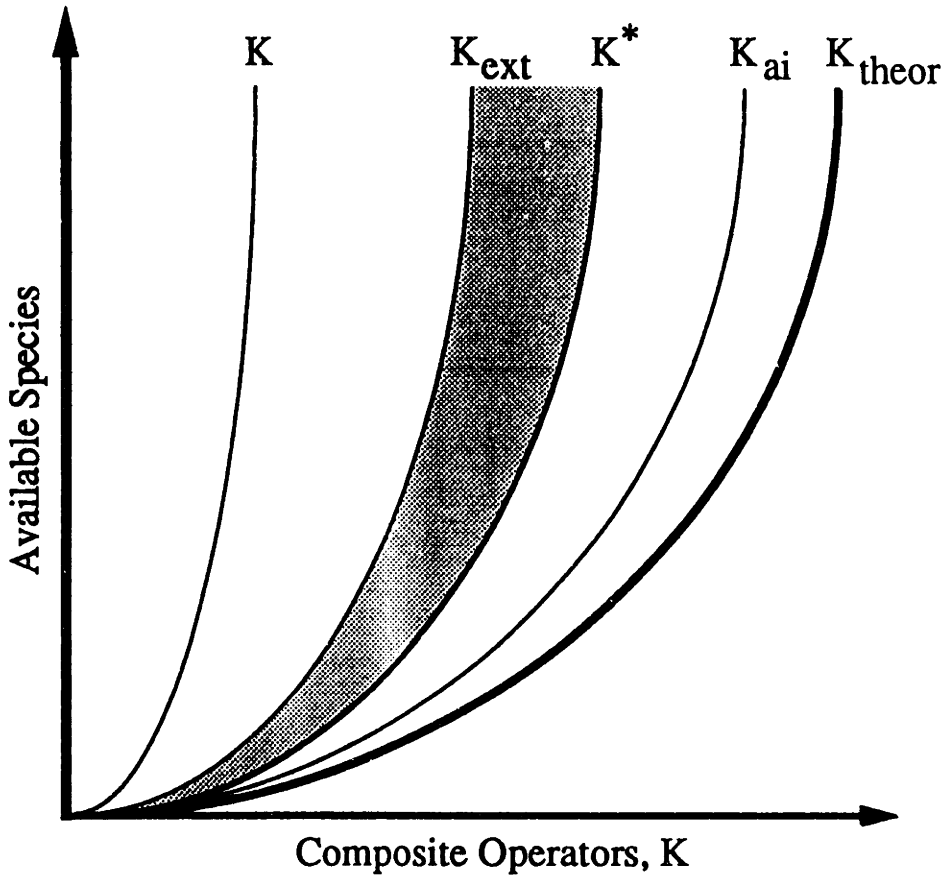


Figure 5-10: Reaction space

responsible for transformation and the supporting **reaction-environment**. **Reactions** are the building blocks for pathway construction.

definition: A pathway, P, is a sequence (i.e. more than one) of reactions that connect an initial reactant or set of reactants to a final product or set of products.

that is,

$$\{abc\} \xrightarrow{K_1} \xrightarrow{K_2} \xrightarrow{K_n} \{abc_n\}$$

is equivalent to,

$$R1 + R2 + \ldots R(n-1) + R_n$$

or more precisely,

$$\{abc\} \xrightarrow{P} \{abc_n\}$$

Notice that all transformation operations require the same information: an **abc** and a transformation operation. Only the level of detail distinguishes ab initio operators, composite operators, reactions, and pathways. As a consequence new operators or meta operators may be built on top of these operations and invoked directly (Fig. 5-11).

For example, if we let $\mathbf{R_1}$ and $\mathbf{R_2}$ describe the relevant transformations of A to I and I to P, as shown below, a new reaction $\mathbf{P^*}$ can be abstracted representing the transformation A to P. Properties of the modeling language (e.g. transitivity) allow the system to understand this new transformation. As a consequence, $\mathbf{P^*}$ may be catalogued and utilized directly in future (pathway) generations.

$$\mathbf{A \rightarrow I \rightarrow P}$$

$$\mathbf{A \xrightarrow{P^*} P}$$

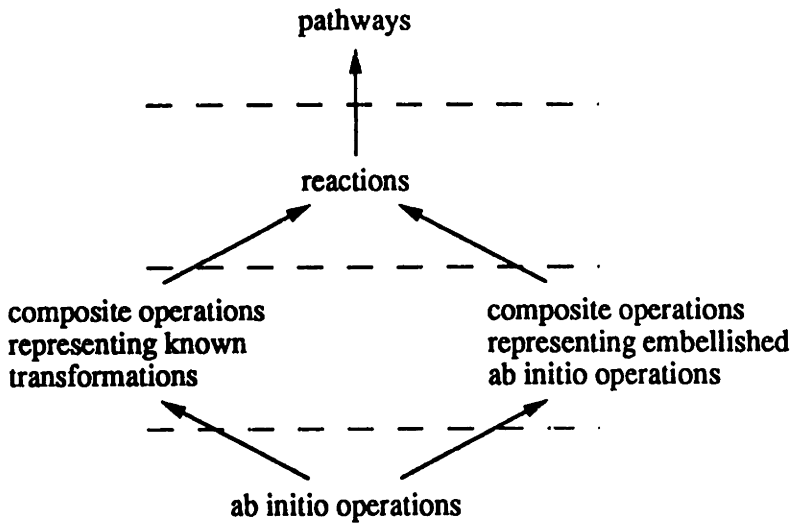


Figure 5-11: Composite operations for pathway generation

This architecture permits dynamic selection of lower lying transformational operations when an appropriate high level transformation is not found (e.g. $\mathbf{K_{ai}}$ selection in the absence of **K**, **R** selection in the absence of P); selection can be user controlled. This provides the most direct generation of pathways - although mechanistically they may be unknown transformations - leading to a particular target.

> **Modeling element 9: context.** A context is a consistent set of assumptions which characterize a species behavior or structural character. If one wants to change the description of some modeling elements (i.e. associate a new set of assumptions with them) but also wants to preserve the former version the need arises for another context.

The use of context is particularly helpful when a reactant can exist in several forms. [Examples include resonance structures, keto enol forms, etc.] If a context is associated with a pathway, assumptions leading to the selection of a particular behavior can be modified and the pathway adjusted accordingly without reinitializing the entire system.

5.7. Subclasses of Model Elements

Of the nine basic modeling elements described in the previous section, four are represented by several classes of modeling object. Each of these classes are characterized by a set of attributes that describe the domain and facilitate chemical reasoning. Although, the modeling elements are general and independent of any specific reaction environment, the classes of modeling objects emanating from them are specific to the domain. In CRL, the classes are designed to elucidate chemical reactivity.

In each case, a multilevel, hybrid representation was chosen to model chemical species and the attending chemical reasoning process. Together with the modeling elements, the representation exposes natural constraints and makes important attributes explicit at the

reasoning level that requires the information[5]. This organization makes possible inclusion of redundant information while maintaining the ability to derive information not explicitly stated; it affords greater modularity and provides additional levels of abstraction that facilitate higher level reasoning [57]. An important feature of classification is that the information associated with a particular class can be inherited to lower classes. A class hierarchy allows the attributes of various classes to be mixed.

The language described in Section 5.8 provides a means for adding classes, attributes, and methods. This organization affords not only evolutionary construction, but it also facilitates the inclusion of increasingly sophisticated procedures to assist in evaluation.

The abc-classes. Four main subclasses emanate from the abc-class. They are: (i) *group*, representing a substructure of the atom-bond-configuration. Additional groups may be added to or built from the list of conventional functional groups (e.g. acid from oxo and alcohol); (ii) *ion*, representing any atom-bond-configuration that has a net electrical charge. (iii) *radical*, representing any atom-bond-configuration that has an open shell, and (iv) *molecule*, representing any atom-bond-configuration that is not an ion or a radical. Further specialization of these primary subclasses, either through the mixing of the primary subclasses (e.g. the class radical-ion may be formed by mixing radical and ion) or by specializing the class description (e.g. creating a class organic under molecule) allows extension of the generic abc-class.

Consider, for example, specialization of the class **molecule** (Fig. 5-12) into **organic-molecule** and **inorganic-molecule** and suppose **inorganic-molecule** contains a method for electron counting. This method is inherited to **organometallic-molecule**; how the method is used and combined with other methods is controlled **organometallic-**

[5]For example, the identification of groups or metagroups within a structure allows those groups to be reasoned about and manipulated directly without having to continuously derive the knowledge germane to the group from the structure itself.

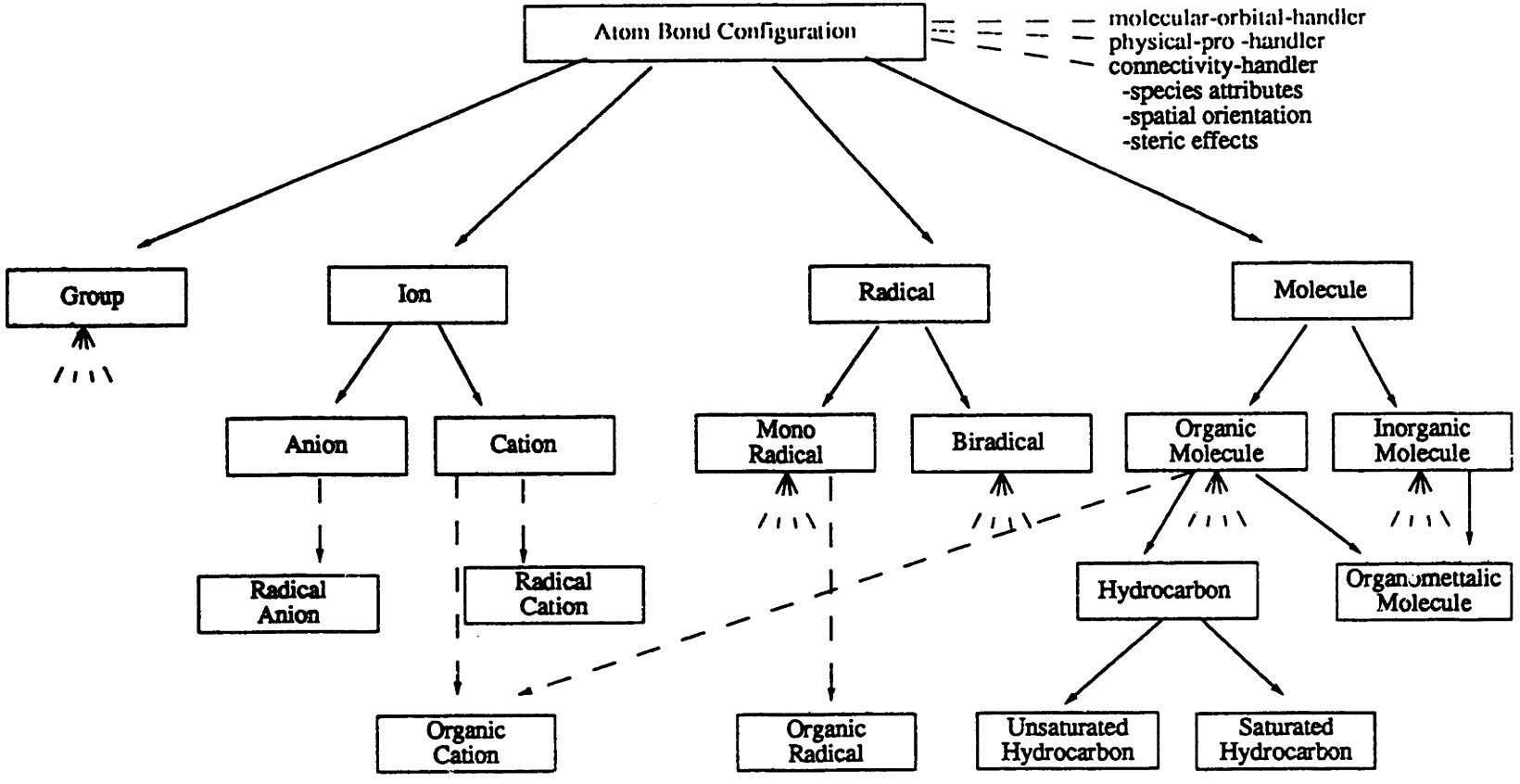


Figure 5-12: Task Organization: chemical structure

molecule. Likewise, if methods and attributes of **organic-molecule** facilitate reasoning about the organometallic system they can be included as well.

The cb-classes. Chemical behavior is classified into two main subclasses: external influences and internal influences (Fig. 5-13). Unlike the abc-class, the cb-class structure provides a means for communicating attribute values between classes, independent of their position within the class structure. This is necessary because chemical behavior is often a combination of effects (e.g. the ground state of a species is influenced by its inherent electronic structure and its external environment). CRL provides a means of combining these effects. These operations comprise the basis for reactivity assignment. They are distributed throughout the hierarchy and are used to classify electronic states. The classes composing this hierarchy categorize electronic state and not the derived properties associated with an electronic state. Nucleophilicity, for example, is derived from the class **pertinent-occupied-molecular-orbital, POMO**; radical behavior is derived from the class **singly-occupied-molecular-orbital, SOMO.**

As in the **abc** hierarchy, each class composing the chemical behavior hierarchy contains only those methods and attributes that pertain to it (Fig. 5-14). Concepts derived at a particular class are used in classes of higher specialization so that more sophisticated concepts can be deduced and discriminating properties elucidated at the proper level. For example, the class **POMO** contains methods for evaluating the pertinent occupied molecular orbital whether it be the HOMO, the *n*-HOMO, or any other occupied molecular orbital. These attributes may then be used at more specialized level to expand on the features of an electronic state which gives rise to a particular conceptual behavior. Nucleophilicity illustrates this point because it may arise from either a negatively charged ion or a neutral atom (i.e. nucleophilicity can result from lone pairs that are bonded, as in the case of the lone pairs on nitrogen, or non-bonded as in the case of the electrons comprising a π bond).

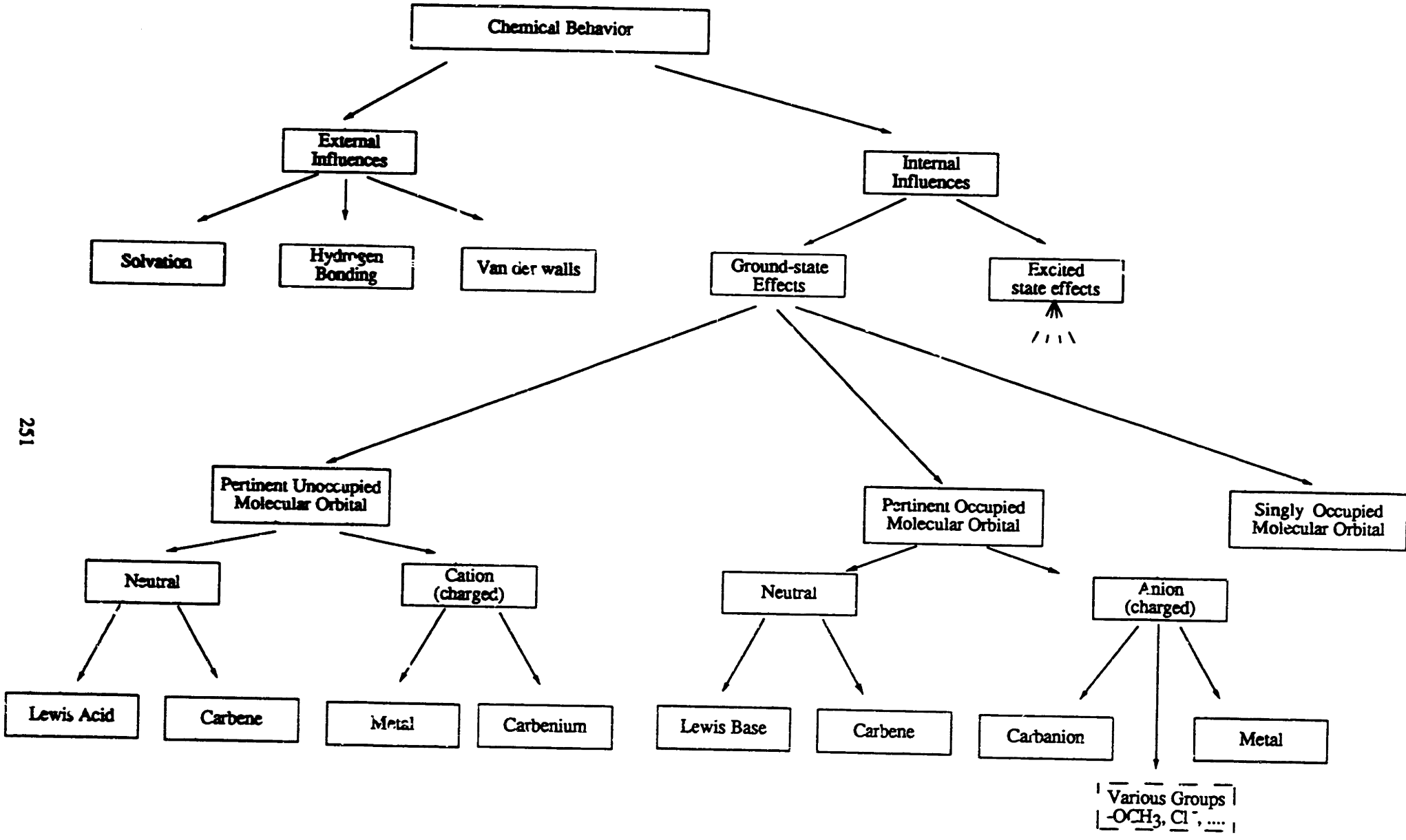


Figure 5-13: Task Organization: chemical behavior

singly occupied molecular orbital attributes

identifier
bonding-orbital
antibonding-orbital
frontier-groups
hyperconjugation
beta-proton-coupling
resonance-structures
radical-stability
reaction-centers
radical-atoms
persistent-radical-p
transient-radical-p
reactivity-index
coupling-constant
...

Figure 5-14: Select attributes of singly occupied molecular orbital

An important distinction between the **abc** and the **cb** hierarchy is that the **cb** hierarchy is not treated as task specialization by operators calling for the assignment of chemical behavior. Since **cb** is a set of potential behaviors, {b}, it is necessary to associate behaviors to a species that lie on the same class level (e.g. **PUMO** and **POMO**) as well as on different, more specialized, levels of the same class (e.g. **Lewis base** and **POMO**). This is necessary because the species dynamically assumes the requisite behavior - (weak) nucleophile or **Lewis base** - when it is required by $K_{get\text{-}sites}$.

The ab-initio operator classes. Five primary subclasses were developed to model transformations (Fig. 5-15); they are: (i) bond formation, (ii) bond cleavage, (iii) single electron transfer, (iv) electron excitation, and (v) electron decay. These operations stem from two types of operations: mass transfer and energy transfer. Each of the subclasses contain methods which prevent the generation of theoretically infeasible structures. As in the abc-class, the subclasses of ab-initio operations can be mixed, together with their methods, to form various new classes. For example, the subclass heterolytic bond cleavage is formed by combining the parent class's of single electron transfer and bond cleavage. Together, these operations afford the generation of any elementary reaction mechanism.

The composite-operator classes. A hierarchy illustrative of **K** is shown in Figure 5-16; like $\mathbf{K_{ai}}$, it is a task organization hierarchy involving specialization. As suggested earlier, composite operators are built upon $\mathbf{K_{ai}}$; they utilize spatial operators, $\mathbf{K_{so}}$, which modify the orientation of an **abc**. Although many hierarchies can be developed utilizing various classification schemes, for efficiency reasons, we have chosen a two-staged classification built upon particle reaction requirements and specialized by structural reaction requirements. Due to this formulation, there are only two primary subclasses extending from composite operator: multimolecular operations and unimolecular operations. Each subclass is then further specialized according to the

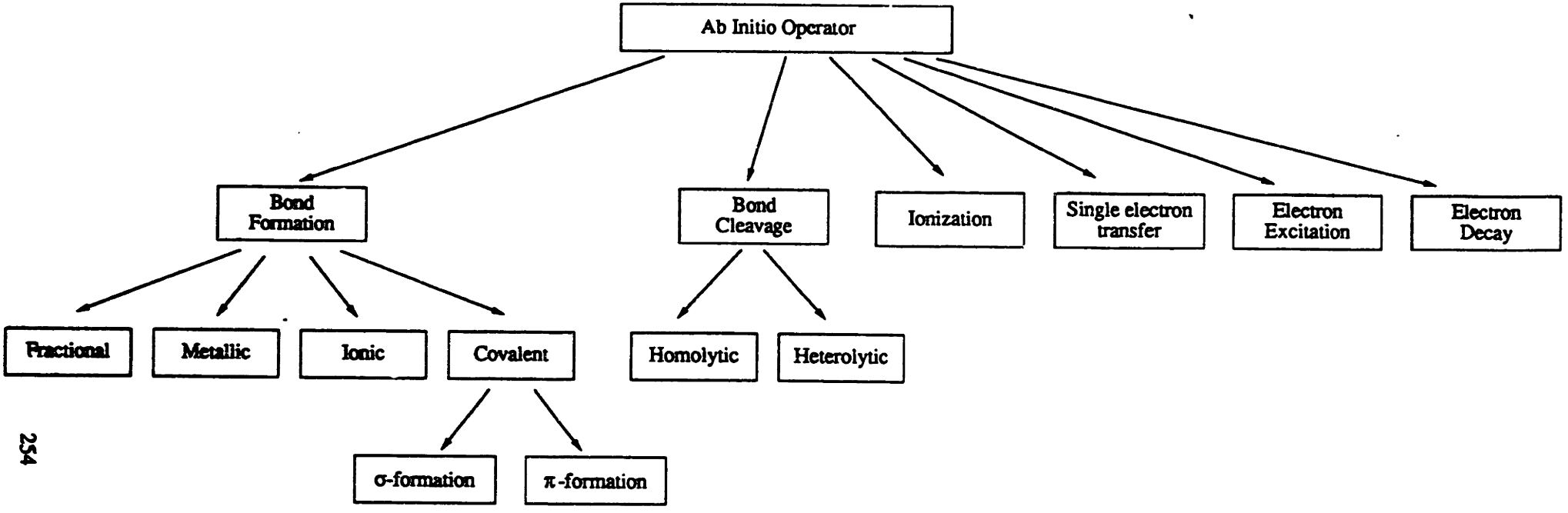


Figure 5-15: Ab initio hierarchy

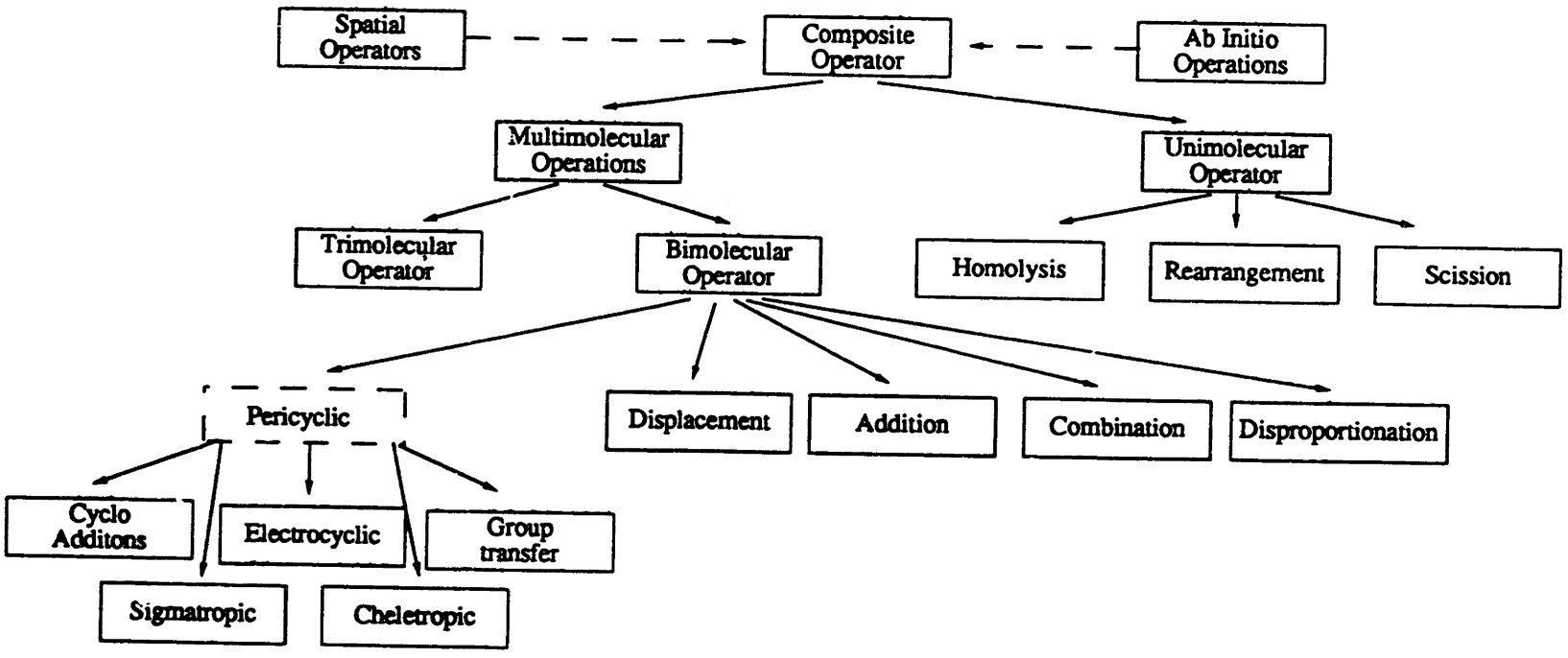


Figure 5-16: Task Organization: composite-operator

structural requirements of the composite operator. For example, the bimolecular operation of radical coupling requires two radical substrates.

This classification limits the combinatorial explosion encountered in conventional computer-aided synthesis because composite operators know a priori when they apply. For example, the application of bimolecular operators to a list of chemical species results in the selective application of **radical-operator** to radicals contained in the list and **molecule-operator** to species that are molecules.

Temporal classifications are also used to reduce combinatorial complexity, since the application of **K** to an **abc** is meaningful only during the life of the reactive/energized species (**abc**). Consequently, alternatives that require more time than the life of the reactive species can not occur and therefore need not be evaluated. Further classification acts to reduce the combinatorial complexity of the problem domain. A good classification scheme, utilizing knowledge contained in the **composite-operator**, will restrict infeasible evaluations.

5.8. Basic semantic relationships among classes of modeling objects

Each of the basic modeling elements, described in the previous section, with the exception of context represents one or more classes of modeling objects. Our approach to the structuring and modularization of knowledge contained is these classes is one of the many approaches afforded from the work on semantic networks. The arrangement of knowledge, in which nodes represent concepts and links connect related concepts (forming a network) is referred to as a semantic network. This representation language, like KL-ONE [58] or KANDOR [59] uses the objects and links between them as parts of the structure of descriptions. Objects have no assertional import. They serve as descriptions of some grouping of knowledge or typing of individuals. Even uninterpreted this type of data structure is undeniably useful for classifying and

managing knowledge derived from complex domains. For the purposes of knowledge representation, however, an interpretation must be imposed on the representational objects - that is they must have *meaning*. This interpretation is referred to as the *language's semantics*. Semantics is "the scientific study of the relations between signs or symbols and what they denote or mean." But there are an extraordinary number of possible interpretations of the links and nodes in a semantic network. Sometimes the exact semantic interpretation is left to the reader. We want to avoid such ambiguities and to provide:

1. precise machine-independent semantics.

2. unambiguous specification techniques.

3. a theory to support implementation independent reasoning.

Classes of modeling objects are often related in a variety of ways, all of which can be formalized as mappings [60] from one class to another. The types of mappings needed for modeling purposes define the set of requisite semantic relationships among the different classes of modeling objects. Of interest to our modeling system are the following types of mappings:

Specialization

Establishes the links between a basic modeling element and the derivative classes of modeling objects. This is an isomorphic type of mapping.

> **Semantic-Relationship-1: "is-a".** From the numerous acceptations that this link has, we only use the Generic/generic type of interpretation [61]. This semantic relation is used to indicate *Subset/Superset* links or *Conceptual Containment* and is needed to define various classes of modeling objects. Each new class of modeling object is a structure

isomorphic to the modeling element that it was derived from, but with more specializations. For example, if

molecule describes the class of molecules

and

abc describes the class of atom-bond-configurations

by using the is-a construction, we can establish a relation like

molecule is-a abc

whose meaning is self-explanatory. The same is true for any link in a specialization hierarchy. One of the fundamental properties of this relation is transitivity. This property allows us to conclude that

hydrocarbon is-a molecule

from hydrocarbon is-a organic-molecule and organic-molecule is-a molecule and makes possible the mechanism of inheritance. The utility of inheritance is that it provides us a means of developing a specialization taxonomy to help us organize knowledge while allowing each subset in the hierarchy to access generic knowledge which lies above it.

Semantic-Relationship-2: "is-a-member-of". This Generic/Individual type of relationship is also referred to as "Set Membership" [61]. It is needed to relate isomorphic structures, emanating from the same class of modeling objects, with identical specialization. For example, assume that specific assumptions on the modeling of free radicals lead to the class of modeling objects called radical. Then if specific free radicals, radical-1 and radical-2, are modeled under the same set of assumptions, the

resulting models will have the same internal structure as radical but different assigned names and values to the various variables or attributes describing it. For example, if

Radical-1 describes the individual entity or instance "Radical-1"

and

Radical describes the class of free radicals

then the construct

Radical-1 **is-a-member-of** Radical

establishes a set membership relation. The same is true for Radical-2. Similarly, molecule-1 is-a-member-of the class molecule. The is-a-member-of semantic relationship allows the generation of models for specific chemical systems, which are isomorphic structures but can be distinguished from each other.

Specification

Specification mappings, described by the following six semantic relationships, are used to specify the value of attributes of various modeling classes. They express (i) binary relations such as whole/part links, (ii) communication lines among modeling objects, such as atom/abc or atom/bond associations, or (iii) the value of simple describing properties.

Semantic-Relationship-3: "is-composed-of". This relationship defines the link between a modeling object and other modeling objects in which the former is decomposed. For instance, to represent a free-radical-

reaction one may want to break it down to the level of Initiation-Reactions, Propagation-Reactions, and Termination-Reactions. Thus,

Free-radical-reaction **is-composed-of** Initiation-reactions

the same is true for Propagation-reactions and Termination-reactions:

Free-radical-reaction **is-composed-of** Propagation-Reactions

Free-radical-reaction **is-composed-of** Termination-Reactions

It is easy to infer that this is a one-to-a-set mapping that has the property of transitivity. For example, by specifying that

Propagation-Reactions **is-composed-of** Elementary-Reactions

the semantic relation is established each time the attribute "components" of a modeling object (e.g. reaction) is given one or more values (e.g. Initiation-Reactions, Propagation-Reactions, Termination-Reactions).

Semantic-Relationship-4: "is-part-of". This relationship defines the link between a modeling object and the modeling object that is containing it. Continuing with the example presented above, we can specify that

Initiation-Reaction **is-part-of** Free-Radical-Reaction

In particular, this semantic relation is established when the attribute "parent" of the unit Initiation-Reaction receives the value Free-Radical-Pathway. The "is-part-of" relationship deals with structural containment - the "is-a" relationship describes conceptual containment. It is obvious that one of the properties of the relation is transitivity, and that "is-composed-of" and "is-part-of" are symmetric relationships. Once one of

these links is established the corresponding inverse is automatically defined. For instance, the equation rate-expression-1 (the rate expression associated with the Free-Radical reaction-1 is-composed-of the terms rate-coefficient, reacting-species, and partial-orders of the reaction with respect to the reacting species. Each of these terms knows that it is-part-of the equation rate-expression-1.

Semantic-Relationship-5: "is-attached-to". To facilitate structure elucidation, semantic relationships 5 and 6 are established to aid reasoning about connectivity. The first, is-attached-to, is designed to reflect atom connectivity. For example, given an atom, oxygen-1 (describing a particular instance of an oxygen atom) and its neighbor atoms (a list of atoms; for example, carbon-1 and carbon-2) the following semantic relation is defined:

oxygen-1 **is-attached-to** (carbon-1 carbon-2)

This link determines the flow of information between connected atoms; the type of information that flows is determined by the type of bond establishing the connection (e.g. covalent, ionic, metallic, fractional). In the example above, the relationship is established when the attribute neighbor-atoms of oxygen-1 receives as value the list carbon-1 and carbon-2. Additionally, since atoms know their membership in various groups, auxiliary links are established using this relationship.

oxygen-1 **is-attached-to** oxo-1

oxygen-1 **is-attached-to** carboxyl-1

oxygen-1 **is-attached-to** meta-group-1

Semantic-Relationship-6: **"is-connected-by".** This is the symmetric relation of is-attached-to. Together these relations allow connectivity reasoning to propagate through atoms, bonds, or both. Following with the previous example and now considering the bonds associated with oxygen-1, bond-1 and bond-2, we can define the following semantic relationships:

oxygen-1 **is-connected-by** (bond-1 bond-2)

From these the system knows that

oxygen-1 **is-connected-by** carbon-1 by bond-1

and

oxygen-1 **is-connected-by** carbon-2 by bond-2

In this example, the relation is established because the attributes "neighbor-atoms" of oxygen-1 and "comprising-atoms" of bond-1 have as a value an carbon-1. As with any symmetric pair of semantic relationships, once one of the links is defined, the corresponding inverse one is automatically established.

Semantic-relationship-7: **"is-described-by".** At a basic level, the state of an object (e.g. atom-bond-configuration, composite-operator, chemical-behavior) is captured by variables. Moreover, its function or behavior is described by relationships. In general, we can state that any object that is a member of the primitive classes is-described-by mathematical components. For example, the atom oxygen-1 and its hybridization are related by

oxygen-1 **is described-by** hybridization-oxygen-1

The attribute hybridization of the atom oxygen-1 establishes a relationship between the atom and its hybridization. The same is true of the other attributes that describe oxygen-1 (e.g. formal charge, valence-electrons, etc.). Along the same lines a transformation, like S_n2, is-described-by relationships of different types (equations, order-of-magnitude, rules, etc.) defining semantic relations such as

S_n2 **is-described-by** K-get-sites-S_n2

S_n2 **is-described-by** K-transformation-S_n2

S_n2 **is-described-by** K-reaction-condition-S_n2

.....

Semantic-Relationship-8: "is-describing". This is the inverse relation of is-described-by. Continuing with the examples presented above, and now considering the mathematical components, we can define the following inverse semantic relationships

Formal-charge-oxygen-1 **is-describing** oxygen-1

K-transformation-S_n2 **is-describing** S_n2

As before, appropriate mechanisms keep these symmetric links properly updated, provided that one of them is defined.

Let us recall that the modeling element *Context* contains the consistent set of assumptions associated with the modeling scope of a pathway. Given that some of these assumptions are used to activate Semantic-Relationships 3 to 8, it is clear that all "specification"-type mappings,

described by these semantic relationships, hold within a given context. Finally, Semantic-Relationships 3, 5-8 have been defined to obey the axioms of commutativity and merging.

Abstraction/Decomposition/Disaggregation

Abstraction is the process by which details of an object are forgotten yielding a structure appreciated for its dominant characteristics. These structures are developed by analyzing the attributes which specify it so that common relevant attributes can be grouped together. This is done in hope of simplifying the problem. Consequently, the process of abstraction can be envisioned as an application of many-to-one mapping. Abstraction is thus characterizable as a mapping of objects of one class into a second presumable less complex class [60]. Disaggregation is just the opposite process; it is the introduction of more details or a refinement of the defining class. Decomposition is the process of factoring a large problem into separable subproblems in such a way that:

1. each subproblem is at the same level of detail.
2. each subproblem can be solved independently.
3. the solutions to the subproblems can be combined to solve the original problem. The goal in decomposition is to create modules or objects that interact with one another in simple, well-defined ways.

Semantic-Relationship-11: "is-disaggregated-in". This semantic relation exists between modeling elements located in different contexts and responds to the need of breaking down systems into smaller, more tractable components, where additional detail can be added. For example, a theoretically feasible but yet unsubstantiated pathway can be characterized by a set of assumptions that lead to the structural character defining the comprising intermediates. This same pathway may be

characterized by another set of assumptions, at a later stage, that leads to a different set of intermediates. Different contexts allow us to encapsulate the distinct representation. However, if we seek to transfer information from the pathway to the species in which it is broken we have to create a communication route. The semantics links

Global-Pathway-1 **is-disaggregated-in** Initiation-Pathway-1

Global-Pathway-1 **is-disaggregated-in** Propagation-Pathway-1

Global-Pathway-1 **is-disaggregated-in** Termination-Pathway-1

provide the communication vehicle.

Notice that the is-disaggregated-in link is designed to support communications between objects of the *same type* in *different contexts*. In this way it is differentiated from is-composed-of. The scope of is-composed-of is restricted to objects of *any type* provided they are in the *same context*. The behavior of is-disaggregated-in is similar to that of is-composed-of with respect to the properties which it supports. This semantic link obeys the axioms of transitivity, commutativity, and merging.

Semantic-Relationship-12: "is-abstracting". The inverse relation of is-disaggregated-in is found to be is-abstracting. The treatment of this relationship is similar to other symmetric relations. Following with the previous example and now considering Initiation-Pathway-1, Propagation-Pathway-1, and Termination-Pathway-1 we can state that

(Initiation-Pathway-1, Propagation-Pathway-1,

Termination-Pathway-1) **is-abstracting** Global-Pathway-1

This semantic link should not be confused with the is-part-of relation. As in the relation is-disaggregating, is-abstracting relates modeling objects located in different contexts producing a many-to-one mapping. Is-part-of maps one-to-one an object in the same context. This semantic relation obeys the axiom of transitivity.

Miscellaneous Semantic Relationships

Semantic-Relationship-13: "is-characterized-as". Although this semantic link does not represent a mapping between classes of modeling objects we have included it because it serves the purpose of specializing a class by describing some of its properties. It represents the semantic relation between a modeling object and an attribute of its description. For example, if the class Pertinent-Occupied-Molecular-Orbital is described as HOMO, and nucleophilicity, semantic links like

POMO is-characterized-as HOMO

POMO is-characterized-as nucleophilic

are established between the modeling objects and the attributes of its description. This semantic relationship is particularly important in defining the context of a particular pathway. For example, after a pathway is constructed the user may wish to change the characterization of an intermediate (e.g. Lewis-acid versus nucleophile) so that the implication of the change can be investigated. This is accomplished easily

with the semantic relationship is-characterized-as. The axioms of commutativity and merging are supported by this relation.

Although all object-oriented systems have, by definition, the following semantic relationships, they have been included here for completeness. Together, these relationships provide a means for describing an object through its attributes and a means for tailoring the process which evaluates an objects value. [For more information on this topic, see reference 62.]

Semantic-Relationship-14: "is-attribute-of". This relationship enables the association of an attribute to an object. Without such a relationship, an object would remain the conventional low-level data structure: it would not be capable of taking on any physical significance. The failure to associate physical significance with an object forces the developer to reason about lower level entities since concepts, represented as objects, no longer can be thought about directly or manipulated directly.

For example, the object representing hydrocarbons may be described, in part, by the addition of the following attributes.

hydrogen-atoms **is-attribute-of** hydrocarbon
carbon-atoms **is-attribute-of** hydrocarbon
aliphatic-p **is-attribute-of** hydrocarbon
aromatic-p **is-attribute-of** hydrocarbon

Semantic-Relationship-15: "is-method-of". Methods are procedures which operate on the objects with which they are associated. They are used to determine the attribute's value or to provide additional information, through evaluation, about an object. Additionally, the

association of class specific methods with an object facilitates bookkeeping as demonstrated below. The first two methods associated with the object hydrocarbon are used to supply attribute values; the latter because it is reasonable to restrict the use of the procedures to that object class. The association of methods with an object can ameliorate the organizational difficulties experienced complex domains.

compute-aliphatic-p **is-method-of** hydrocarbon

compute-aromatic-p **is-method-of** hydrocarbon

collect-longest-carbon-chain **is-method-of** hydrocarbon

carbon-chain-length **is-method-of** hydrocarbon

5.8.1 Representation of modeling objects - the entity attribute set

As it was mentioned earlier, all the modeling elements that belong to the taxonomies of the abc, atom-bond-configuration, chemical-behavior, composite-operator, and ab initio operator are represented by objects. Slots within objects portraying such modeling elements are used to represent different sorts of attributes, as well as relationships among the several components of a model. The following examples will illustrate these concepts.

For instance, an object that is representing a chemical structure (e.g. group), is described in terms of physicochemical assumptions and phenomena. The characterization is typical of objects that belong to the taxonomy of atom-bond-configuration. As mentioned earlier, methods associated with the object modify the values of the attributes describing the object. Methods which operate on the object but which do not modify the values of describing attributes are also associated with the object. For example, collect-sp^2-hybridized-atoms is a method associated with abc that returns all sp^2 hybridized atoms belonging to a particular abc. More specifically, this object may show that group is-

connected-by bond-1 and is-characterized-by hydrocarbon, aliphatic, alkene, nucleophile, base, etc. The attributes of an object should be focused on allowing a complete articulation of the models' underlying behavior, thereby, documenting the hypotheses and simplifications associated with the model. Some of these attributions will play an important role during the process of instantiation of the model; they will allow the automatic generation of the basic constraints associated with the model instances.

Figure 5-17 shows an object describing the particular features of an instance of organic-molecule named ethane-1. Because ethane-1 is-a-member-of organic-molecule its structure is the same as that describing organic-molecule. However, the attributes that account for the specification type of relations now have as values names of specific instances. For example,

methyl-1 **is-part-of** ethane-1

ethane-1 **is-a-member-of** organic-molecule

methyl-1 **is-connected-by** bond-1

The examples presented above show how objects incorporate the semantic relationships discussed in Section 5.2.3. The underlying language used to implement this system (Symbolics common Lisp with Object-Oriented Extensions) provided the capabilities to relate objects by means of "**is-a**" and "**is-a-member-of**" semantic relationships, while the "specification" and "abstraction/disaggregation" type of relations had to be implemented as part of the development of the language. The "specification" semantic relationships are the links that allow us to represent a system as a *composite object.* A composite object is a group of interrelated objects. Composite objects in this environment explicitly capture and enforce semantic relations 3 to 8. They augment the semantic integrity of an object-based model through the notion of *dependent object.* A dependent object is one

```
identifier:                    "C2H6-T0779"
name:                          "ethane"
atoms:                         (#<ATOM C-T0764> #<ATOM H-T0765>
                                #<ATOM H-T0766> #<ATOM H-T0767>
                                #<ATOM C-T0768> #<ATOM H-T0769>
                                #<ATOM H-T0770> #<ATOM H-T0771>)
bonds:                         (#<BOND b-T0772>  #<BOND b-T0773>
                                #<BOND b-T0774>  #<BOND b-T0775>
                                #<BOND b-T0776>  #<BOND b-T0777>
                                #<BOND b-T0778>)
empirical-formula:             ((#<DB-ATOM 414000575> . 6) (#<DB-ATOM 414001073> . 2))
empirical-formula-string:      "C2H6"
molecular-weight:              30.07
charge:                        0.0
terminal-skeleton-atoms:       (#<ATOM C-T0764>  #<ATOM C-T0768>)
equivalent-atoms:              ((#<ATOM C-T0768> #<ATOM C-T0764>)
                                (#<ATOM H-T0771> #<ATOM H-T0765>
                                #<ATOM H-T0766> #<ATOM H-T0767>
                                #<ATOM H-T0769> #<ATOM H-T0770>))
equivalent-bonds:              ((#<BOND b-T0775>)
                                (#<BOND b-T0778>     #<BOND b-T0772>
                                #<BOND b-T0773>     #<BOND b-T0774>
                                #<BOND b-T0776>     #<BOND b-T0777>))
weakest-bond:                  (#<BOND b-T0775>)
weakest-bond-strength:         82.6
weakest-bond-strength-ratio:   1.0
ordered-eq-bonds:              ((#<BOND b-T0775>)
                                (#<BOND b-T0778>     #<BOND b-T0772>
                                #<BOND b-T0773>     #<BOND b-T0774>
                                #<BOND b-T0776>     #<BOND b-T0777>))
groups:                        (#<GROUP methyl-1>  #<GROUP methyl-2>
                                #<GROUP terminal-sp3-methylene-1>
                                #<GROUP terminal-sp3-methylene-2>)
group-bonds:                   (#<BOND-b-T0775>     #<BOND b-T0772>
                                #<BOND b-T0776>)
meta-groups:                   (#<GROUP ethyl-1>     #<GROUP ethyl-2>)
methyl-carbon:                 NIL
primary-carbons:               (#<ATOM C-T0764>  #<ATOM C-T0768>)
secondary-carbons:             NIL
tertiary-carbons:              NIL
terminal-carbons:              (#<ATOM C-T0764>  #<ATOM C-T0768>)
backbones:                     (#<ATOM C-T0764>  #<ATOM C-T0768>)
backbone-length:               2
progenitor:                    NIL
environment:                   #<reaction-environment-1 11235723>
  .                              .
  .                              .
  .                              .
```

Figure 5-17: An instance of ethane

whose existence depends on the existence of another object. For instance, methyl-1 is itself a composite object. In consequence, it makes no sense to consider the bond, bond-1, if the bond is not "**attached-to**" methyl-1. However, this does not imply that it is senseless to reason about methyl-1 ignoring its specific context. These examples show that a constituent object (except the root) of a composite object cannot be created unless the unit on which it depends already exists. Furthermore, when a constituent object of a composite object is deleted, all its dependent objects are also deleted. These ideas are particularly important during the processes of definition of new composite objects, as will as modification or deletion of existing ones.

5.8.2 Properties of CRL

The semantic relations of CRL establish how different objects relate to one another; they were formally defined to obey the following axioms:

Axiom 1 -- Transitivity

If (O1 "*semantic-relation*" O2) and (O2 "*semantic-relation*" O3)

then

(O1 "*semantic-relation*" O3)

where "*semantic-relation*" applies to the relations **is-a**, **is-composed-of** and **is-part-of**, and O1, O2 and O3 are arbitrary modeling objects of this system.

For example,

If (Methyl-1 IS-PART-OF Ethyl-1) and (Ethyl-1 IS-PART-OF Ethane-1)

then

(Methyl-1 IS-PART-OF Ethane-1)

Axiom 2 -- Commutativity

(THE "*attribute-1*" OF O1 IS A_1)

. . . . . .

(THE "*attribute-1*" of O1 IS A_{j-1})

(THE "*attribute-1*" of O1 IS A_j)

. . . .

(THE "*attribute-1*" of O1 IS A_n)

is the same as

(THE "*attribute-1*" of O1 IS A_1)

. . . . . .

(THE "*attribute-1*" of O1 IS A_j)

(THE "*attribute-1*" of O1 IS A_{j-1})

. . . . . .

(THE "*attribute-1*" of O1 IS A_n)

where *attribute-1*" is any specific attribute establishing one of the following semantic relationships: **"is-composed-of"**, **"is-attached-to"**, **"is-connected-by"**, **"is-described-by"**, **"is-describing"**, and **"is-characterized-as"**. In other words, the order in which attributions are listed is irrelevant.

Axiom 3 -- Merging

(THE "*attribute-1*" OF O1 IS A_1)

. . . . . .

(THE "*attribute-1*" of O1 IS A_{j-1})

(THE "*attribute-1*" of O1 IS A_j)

. . . . . .

(THE "*attribute-1*" of O1 IS A_n)

same as

(THE "*attribute-1*" of O1 IS SET.OF ($A_1 \ldots A_{j-1}\ A_j \ldots A_n$))

where "*attribute-1*" is any specific attribute establishing one of the following semantic relationships: "**is-composed-of**", "**is-attached-to**", "**is-connected-by**", "**is-described-by**", "**is-describing**", and "**is-characterized-as**". Hence, attributions of the same concept can be merged.

For example,

(THE Neighbor-atoms of Carbon-1 IS Hydrogen-17)

(THE Neighbor-atoms of Carbon-1 IS Nitrogen-21)

(THE Neighbor-atoms of Carbon-1 IS Carbon-2)

is the same as

(THE Neighbor-atoms of Carbon-1
IS THE SET OF (Hydrogen-1, Nitrogen-21, Carbon-2))

5.9. Syntax

Grammars are intended to capture what we may call the *structure* or *syntax* of languages, as opposed to their meaning or semantics. The syntax of a programming language is a strict, precise set of rules which describe the string of symbols that constitute legal statements and which specify how a statement breaks down into its constituent parts. The syntax description is done using formal grammar, often referred as metalanguage (i.e. a special language used to describe other languages). The description language used here is an extended BNF (Backus-Naur-Form or Backus-Normal Form) which is more compact than BNF. The Backus-Naur Form [63] is a widely used formal method developed by computer scientists for the precise syntactic description of computer languages. The BNF grammars conventionally have four elements: (i) a *terminal vocabulary* which corresponds directly to the vocabulary of allowed tokens of the language to be defined; (ii) a *nonterminal vocabulary*, conventionally enclosed in pointed brackets (< and >); (iii) a *set of production rules* which defines way of building up phrases (represented by nonterminal symbols) from terminal and nonterminal vocabulary

items, and (iv) a *correspondence* indicating which nonterminal symbol is associated with the "master" or "sentence" -phrase type of the language. In defining the grammar we will follow the standard practice of writing a production beginning with the "master" -phrase type at the top of the grammar.

Two important observations can be made about BNF grammar rules [64]. First, BNF rules can be defined in a recursive manner; that is, the phrase name can appear on both sides of the symbol"::=". This property can be observed in the rules defining "<structural-def>", "<input-def>", "<output-def>", etc. This recursive property allows an infinite number of statements to be described by a finite number of rules. Second, a BNF description of a grammar is complete when every phrase name has been defined reaching the level of terminal vocabulary.

CRL's Syntax

Explanation of notation used.

::= : Is defined as.

[] : Encloses optional unit.

{} : Indicates mandatory choice.

<> : Unit which is described separately.

.... : Indicates repetition of syntactic signs.

| : Separator for alternatives.

* : What the braces enclose may appear any number of times (including zero).

+ : What the braces enclose may appear any non-zero number of times (must appear at least once).

[[]] : Double brackets indicate that any number of the alternatives enclosed may be used, and those may occur in any order, but each alternative may be used at most once unless followed by a star.

Word in capital letters: Reserved word.

Lower case words: Metalinguistic variable names.

<modeling-element> ::= <modeling-element-name>[<attribute>]*.

<modeling-element-name> ::= (<modeling-class> | <atom> | <bond> | <reaction> |
<reaction-environment> | <context>) [<attribute>]*
[<method>]+.

<attribute> ::= <attribute-name> IS-ATTRIBUTE-OF <modeling-element>.

<method> ::= <method-name> IS-METHOD-OF <modeling-element>.

<class> ::= (<class-name> IS-A {[<modeling-class]+ | [<class>]+}).

<instance> ::= (<instance-name> IS-MEMBER-OF {[<class>]+}.

<abc> ::= <abc-input>.

<abc-input> ::= be-matrix | atom-table bond-table.

<abc> ::= <group>|<ion>|<radical>|<molecule>.

<group> ::= [atom]+[bond]+.

<ion> ::= [atom]+[bond].

<radical> ::= [atom]+[bond].

<molecule> ::= [atom]+[bond]+.

<cb> ::= {[<behavior>]+}.

<behavior> ::= (IS-CHARACTERIZED-BY{<Internal-Influences>}
[<External-Influences>]).

<Internal-Influences> ::= (IS-CHARACTERIZED-BY
{<Ground-state-effects><Excited-state-effects>}).

<Ground-state-effects> ::= (IS-CHARACTERIZED-BY
{<PUMO>|<POMO>|<SOMO>}).

<Excited-state-effects> ::= (IS-CHARACTERIZED-BY {σ* | π*}).

<External-Influences> ::= (IS-CHARACTERIZED-BY {[<solvation>][<hydrogen-bonding>][<Van der Waals>]}).

<composite-operator> ::= <inputs><enabling-conditions><ab-initio-operator> <output> [<composite-operator>]

<input> ::= <react-env><abc><cb>.

<output> ::= <reaction>.

<abc> ::= {[(Instance-of abc)]$^+$}.

<cb> ::= {[(Instance-of cb)]$^+$}.

<reaction-env> ::= {(Instance-of reaction-environment)}.

<enabling-conditions> ::= {[<conditionals>]$^+$}.

<conditionals> ::= [[[(Instance-of user preferences)]$^+$ {[(Instance-of physical requirements)]$^+$}]].

<ab-initio-operator> ::= {[<K_{ai}>]$^+$}.

<K_{ai}-operator> ::= Bond formation | Bond Cleavage | Ionization | Single Electron Transfer | Electron Excitation | Electron Decay.

<K_{ai}> ::= <K_{ai}-operator><K_{ai}-input><K_{ai}-enabling-cond><output>.

<K_{ai}-input> ::= {[<reaction-center>]$^+$}.

<reaction-center> ::= {[<atom>]$^+$}.

<K_{ai}-enabling-cond> ::= {[<K_{ai}-condition>]$^+$}.

<K_{ai}-condition> ::= {[physicochemical-requirements]$^+$}.

<physical-requirements> ::= (IS-ABSTRACTING <physicochemical-requirements>).

5.10. Conclusions

A language for modeling chemical behavior and reasoning about chemical systems, called CRL, has been presented. The specialized modeling language supported by a theoretical framework that enhances the expressiveness of computer aided chemical

reasoning systems thus far advanced. Realizing the limitations of the previous procedural attempts, it is based on an object-oriented, declarative approach. The system is evolutionary in its design; new knowledge may easily be incorporated into the structures as can additional modeling elements. The architecture inherently eliminates many of the problems encountered previously and affords a framework for integrating previous efforts in the field of chemical process development. We believe the current system provides for the first time a unitary framework for the design of chemical systems using a integrated systems approach.

CRL can be viewed as a very high level special purpose language, that moves the user several levels away from the inherent programming language (e.g. Lisp in this case). CRL's characteristics have been extensively discussed; it has been engineered to: (i) discover new synthetic pathways; (ii) decompose smoothly from theoretically bounded reactions to known reactions, (iii) facilitate rapid and easy extension to new reaction classes; (iv) learn to enhance search efficiency; (v) capture and utilize qualitative, semi-quantitative relationships (ordinal, order-of-magnitude), or boolean relationships; (vi) offer explicit documentation of all the hypotheses, assumptions and simplifications that give rise to a particular model, pathway, or reaction.

The construction of models has been formally addressed, with specific references to its grammar and semantics. CRL has an English like syntax: very little training is required to write the specifications of the model class. This feature makes it possible for all potential users of the language to participate actively in the specification of new modeling classes or in the modification of existing ones.

CRL complies with all the requirement that were proposed for its design. Its strengths are its modularity and its inherent capability of controlling complexity by breaking apart complex chemical systems piece by piece, into smaller, less complex pieces. The

specification of a model class involves subsequent specification and characterization of all its components. This idea is recursively applied throughout the definition process. Thus, it is similar to describing a synthetic plan or a complex chemical system in a natural language. Another important feature of this language is its extensibility; the ability to characterize a chemical system can be extended by incorporating a richer terminal vocabulary. Its modularity makes possible the incorporation of new blocks in the definition of a class; blocks that may describe other aspects that are presently not considered. CRL supports the creation of new data types and modeling elements.

The system is currently implemented on a Symbolic 3650. Excluding interfaces and databases, the program consists of approximately 50,000 lines of Lisp code.

5.11. Bibliography

1. SIGART, 1990

2. Lathrop, Webster, and Smith, "ARIADNE: Pattern-Directed Inference and Hierarchical Abstraction in Protein Structure Recognition," *Communication of the ACM*, Vol. 30, No. 11, pp.909, 1987

3. Westerberg et. al. "DECADE - A Hybrid Expert System for a Catalyst Selection - Final Architure and Results," *Comput. chem. Engng.*, Vol 12, No. 9/10, pp.923-938, 1988

4. Stephanopoulos, George, "The Future of Expert Systems in Chemical Engineering," *Chemical Engineering Progress*, September 1987, pp. 44-51.

5. Henning, H., Leone, H., and Stephanopoulos, G, "MODEL.LA. A Modeling Language for Process Engineering, Part I. The Formal Framework," Comp. Chem Engng.,**14**(8) p. 813 (1990)

6. Vernin, G. and M. Chanon, eds., Computer Aids to Chemistry, Ellis Horwood Ltd., Chichester, England 1986.

7. Dugundji, James, and Ivar Ugi, "An Algebraic Model of Constitutional Chemistry as a Basis for Chemical Computer Programs," *Top. in Curr. Chem.* 39(19) (1973)

8. Fornari, Tiziana, Enrique Rotstein and George Stephanopoulos, "Studies on the Synthesis of Chemical Reaction Paths--II. Reaction Schemes with Two Degrees of Freedom," *Chemical Engineering Science*, 44:7, pp. 1569-1579, 1989.

9. R. B. Woodward, in *Perspectives in Organic Chemistry* (A. Todd, ed.), Interscience Pub., NY (1956), pp. 155-160.

10. R. B. Woodward, *Pointers and Pathways in Research,* G. Hofteizer for Ciba of India, Ltd, Bombay, India (1963), p. 23.

11. E. J. Corey, *Pure Appl. Chem.*, 14, 19-37 (1967).

12. E. J. Corey and W. T. Wipke, *Science*, 166, 178 (1969).

13. M. Bersohn and A. Esach, *Chem. Rev.*, 76, 269-282 (1976).

14. "Computer-Assisted Organic Synthesis," (W.T. Wipke and W.J. Howe, eds) *ACS Symposium Series*, 61, Washingtom DC (1977).

15. R. Carlson, C. Albano, L. G. Hammarstom, E. Johansson, A. Nilsson and R. E. Carter, *J. Molecular Science*, 2, 1-22, (1983).

16. A. K. Long, S. D. Rubenstein and L. J. Joncas, *Chem. Eng. News*, 9, 22-30 (1983).

17. D. Pensak, PhD, Harvard University Sept. (1973) University Microfilm International, Ann Arbor, Michigan, USA.

18. E.J. Corey, A.K. Long and S.D. Rubenstein, *Science*, 228, 408-18 (1985).

19. I. Ugi, J. Bauer, J. Brandt, J. Friedich, J. Gasteiger, C. Jochum and W. Schubert, *Angew. Chem. Int. Ed.*, 18, 111-123 (1979).

20. J.B. Hendrickson, E. Braun-Keller and G.A. Toczko, *Tetrahedron*, 37, suppl. 1, 359-70 (1981).

21. M. Bersohn, in 'Computer Assisted Drug Design,' *ACS Symposium, 112* (E.C. Olson and R.E. Christoffersen eds) Washington DC (1979), pp. 341-352.

22. A. Weise, *J. Prakt. Chem.*, 322, 761-68 (1980).

23. H. Gelernter, S.S. Bhagwat, D.L. Larsen and G.A. Miller, in *Computer Applications of Chemistry* (S.R. Heller, R. Potenzone, eds) Elsevier Sci. Publ., Amsterdam (1983), pp. 35-59.

24. R. Barone, M. Chanon, P. Cadiot and J.M. Cense, *Bull. Soc. Chim. Belg.*, 91. 333 (1982).

25. P. Gund, E.J.J. Grabowski, G.M. Smith, J.D. Endosse, J.B. Rhodes and W.T. Wipke, in 'Computer Assisted Drug Design,' *ACS Symposium Series 112*, Washington DC (1979), p. 527.

26. W.T. Wipke, and D. Rogers, *J. Chem. Inf. Comput. Sci.*, 24, 71-81 (1984).

27. J. E. Dubois, *Isr. J. Chem.*, 14, 17 (1975).

28. J.E. Dubois, *Pure Appl. Chem.*, 53, 1313-1327 (1981).

29. C. Laurenco, L. Villien and G. Kaufmann, *Tetrahedron*, 40, 2721-29 (1984) ibid., 2731-40 (1984).

30. C. Laurenco and G. Kaufmann, *Tetrahedron Lett.*, 21, 2243 (1980).

31. T.H. Varkony, D.H. Smith, and C. Djerassi, *Tetrahedron*, 34, 841-52 (1978).

32. W.L. Jorgensen, *Kagaku*, 38, 483-488 (1983).

33. Z. Hippe, *Analytica Chim. Acta*, 133, 677-83 (1981).

33. E.G. Smith and P.A. Becker, *The Wisswesser Line Formula Chemical Notation*, Chemical Information Management Inc., Cherry Hill, NJ (1976).

35. J.J. Vollmer, *J. Chem. Ed.*, 60, 192 (1983).

35. G.M. Dyson, *A New Notation and Enumeration System for Organic Compounds*, 2nd Ed., Longmans, London (1949).

36. *IUPAC Nomenclature of Organic Chemistry* Pergamon Press Oxford (1979).

38. A.T. Balaban, *Chemical Applications of Graph Theory*, Academic Press, London (1976).

39. H. L. Morgan, *J. Chem. Doc.*, 7, 154-165 (1967).

40. G. Moreau, *Nouv. J. Chimie*, 4, 17-22 (1980).

41. Wipke, W. Todd, Stephen R. Heller, Richard J. Feldmann and Ernest Hyde, eds., Computer Representation and Manipulation of Chemical Information, Wiley and Sons, New York, 1974.

42. Le Blanck, J. R., Moore, D. O. and Cover. A. E., "Coal can be Gasoline," *Hydrocarb. Process.*, Vol. 60, pp. 133-137, 1981

43. Rotstein, E., Resasco, D. and Stephanopoulos, G., "Studies on the Synthesis of Chemical Reaction Paths. I". *Chem Engng Sci.*, Vol. 37, pp. 1337-1352, 1982

44. Stephanopoulos, G., Rotstein, E. and Resasco, D., "Synthesis and Screening of Alternative Reaction Paths," *Ind. chem. Engng Symp. Ser.*, No.74, pp.173-185, 1982

45. Stephanopoulos, G. and Townsend, D., "Synthesis in Process Development," *Chem. Engng. Res. Des.*, Vol. 64, pp.160-174, 1986

46. Wipke, W. Todd, "Computer-Assisted Three-Dimensional Synthetic Analysis," in Computer Representation and Manipulation of Chemical Information, pp. 147-174.

47. Meyer, B., "Reusability: The Case for Object-Oriented Design," IEEE Software, March, 50-64 (1987)

48. Stephanopoulos, George, "The Future of Expert Systems in Chemical Engineering," *Chemical Engineering Progress*, September 1987, pp. 44-51.

49. R. J. Brachman and H. J. Levesque, Readings in Knowledge Representation, MOrgan Kaufmann Pubishers, Los Altos, 1985

50. Abelson, Harold, Gerald Jay Sussman and Julie Sussman, Structure and Interpretation of Computer Programs, MIT Press, Cambridge, Massachusetts, 1985.

51. Maher, M. L., "Expert Systems for Structural Design," Expert Systems in Engineering, D. T. Phan, Ed. IFS Publications/Springer-Verlag (1988)

52. Sriram, D. and M. L. Maher, "The Represetation and Use of Constraints in Structural Design," Applications of Artificial Intelligence in Engineering Problems, Volume 1, 1st International Conference Southampton University, U. K., Sriram and R. Adey Eds., April (1986)

53. Sussman, G. and G. L. Steele, "CONSTRAINTS - A Language for Expressing Almost Hierarchical Descriptions," Artificial Intelligence, 14, 1-39 (1980)

54. R.S. Cahn, C.K. Ingold and V. Prelog, *Experientia* **12**,81 (1956)

55. R.S. Cahn, C.K. Ingold and V. Prelog, *Angew. Chem.* **78**, 413 (1966)

56. W.T. Wipke and T.M. Dyott, *J. Am. Chem. Soc.*, 96,4825 (1974).

57. Liskov, Barbara and John Guttag, Abstraction and Specification in Program Development, MIT Press, Cambridge, Massachusetts, 1986.

58. Brachman, R.J. and J. G. Schmolze, "An overview of KL-ONE Knowledge Representation System", *Cognitive Science*, 9, 171 (1985).

59. Patal-Scheneider, P. F., "Small can be Beautiful in Knowledge Representation", Proceedings of the IEEE Workshop on Principles of Knowledge-Based Systems, Denver, Colorado, December (1984).

60. Ziegler, B.P., "Multifacetted Modelling and Discrete Event Simulation", Academic Press, London (1984).

61. Brachman, R.J., "What ISA Is and Isn't: An Analysis of Taxonomic links in Semantic Networks", Fourth National Conference of the Society for Computational Studies of Intelligence, Saskatoon, Saskatchewan, May 17-19 (1982).

62. Keene, Sonya E., Object-Oriented Programming in COMMON LISP, Addison-Wesley, Reading, Massachusetts (1989).

63. Naur, P. Revised Report on the Algorithmic Language ALGOL 60, Communications of the ACM, 6, 1-17 (1963)

64. Wirth, N., "Algorithms + Data Structures = Programs," Prentice-Hall, Englewood Cliffs, NJ (1976)

Chapter 6
A Language for Reasoning about Chemical Reactivity
Part II: Reaction Pathway Generation

Summary

A modeling language (CRL) has been constructed for the interactive or automatic definition of models for chemical reasoning. It is based on nine modeling elements and fifteen semantic relationships obeying basic axioms of transitivity, monotonicity, commutativity and merging. Its syntax can be described by an extended BNF (Backus-Naur-Form). The structure of chemical pathways is depicted by specific digraphs, which are symbolically constructed by algorithmic procedures driven by the context of the chemical environment. The procedures and models utilized in CRL can: (a) view the chemical system at various levels of abstraction; (b) capture qualitative, semi-quantitative relationships, or boolean relationships; (c) offer explicit documentation of all the hypotheses, assumptions and simplifications that give rise to a particular model, pathway, or reaction. Its object-oriented modularity makes it extensible and easily maintainable. Although a large part of CRL is domain-independent, its vocabulary and syntax is specific to the activities involving chemical reasoning. The use of CRL is illustrated through a series of case studies; each highlights a different feature of CRL.

6.1 Introduction

The synthesis of new chemical species, new reaction pathways, and new chemical processes are attractive objectives within the general field of process synthesis [1, 1a-c] and encompass chemical synthesis (e.g. fredrickamycin or 6-amino-penicillanic acid) or process synthesis (e.g. identification of reaction routes to chemicals which exhibit good economic potential using C_1 and oil based chemicals, or of closed cycle reaction systems for energy storage) or both (e.g. identification of new routes and processes which make accessible the production of difficult products such as soda, chlorine from hydrogen chloride, aluminum from aluminum oxide, hydrogen from water, or the electrowinning of metals and chemical reaction heat pumps).

The various synthetic tasks that provide the underpinnings to these activities often proceed through a series of stages involving different assumptions, decisions, and requiring different levels of detail in describing the task at hand. Such hierarchical evolution in synthetic activities is not new; it has always been implicit in the analysis, evaluation, and design of complex systems [1], and has been implemented in several areas like, chemical synthesis and design [2-4], process design [5-10] analysis of circuits[11], and medical diagnosis [12]. For example, during the development of synthetic chemicals, a chemist begins with *the plan*. A successful plan unravels the molecular network of the desired molecule, piece by piece, until simple pieces are obtained: pieces which can be reassembled to construct the objective [13]. Likewise, during preliminary process synthesis the designer begins with a simple, overall description of the process under development [10], evaluates it, and makes various decisions. Subsequently, additional detail is introduced that allows the designer to decompose the overall process into subsystems (e.g. reactor and separator sections).

In both cases, the synthesis of chemical species or the development of chemical processes, as refined descriptions are introduced they require different models (e.g.

synthetic mechanisms), are based on different assumptions (e.g. reaction conditions), and design decisions (e.g. strategy for functional group addition). To automate such a synthetic process, we need to have a facility that can provide *multi-level modeling* of the same plan/process with internal consistency (i.e. logical and quantitative) among the models at the various levels of abstraction used to develop the plan.

Quite often the designer retains several versions of the evolving synthetic plan corresponding to different reaction trajectories and different contexts: these are defined by differing sets of assumptions and decisions. Therefore, a modeling system should be capable of supporting in an automatic or interactive manner a *multiple-context* modeling of the same plan/process. This will allow explicit comparison of alternative pathways and processes and allow systematic back-tracking of strategic decisions. Additionally, during the course of a synthetic activity, one does not need at the same time all the different perspectives included in the defining models such as, structural, topological, quantitative relationships, or occurring physicochemical phenomena. Therefore, the developer should be able to tug on certain strands, specific to their domain of interest (i.e. a perspective), without interfering with other strands.

Such *multi-viewing* of process models combined with multi-level and multi-context requirements define the scope of *multi-faceted modeling* of chemical systems; the subject matter of this chapter. CRL, whose formal framework was developed in chapter 5, is a concise modeling language which allows multifaceted modeling by providing the following facilities:

1. Multiple viewing of chemical systems and synthetic plans in terms of structure, topology, and behavior.

2. Disaggregation of abstract models to more detailed ones and aggregation of detailed descriptions to more abstract.

3. Contextual description of alternative models for the same process.

4. Controlled flow of information among the models at various levels or in various contexts, and detection of modeling conflicts.

This chapter has been organized as follows: Section 6.2 illustrates how information is accessed in the modeling elements; Sections 6.3 and 6.4 present the formal construction of subclass models: atom-bond-configuration, chemical behavior, composite operator, and ab initio operator; Section 6.5 shows how the semantic relations of abstraction and disaggregation are used; Section 6.6 presents several illustrations; Section 6.7 provides a summary of the language's characteristics.

6.2 Accessing Information

Models generated with CRL are rich. They enclose a significant amount of information stored as attribute values in the objects that are representing the different model components (see chapter 5). These components may be complex objects with their own internal structure. They are linked by specific semantic relations. This section will present mechanisms available in CRL for retrieving specific information from a model. CRL allows access to the different model views in a straightforward manner, without having to deal with the details of the model's internal representation.

6.2.1 Accessing the mathematical components

A CRL user may want to focus attention only on the mathematical components of a model. These components, identified as:

$$\{\text{Mathematical components of } x_i\} = \{x_j / x_j \text{ is-describing } x_i\}$$

include the mathematical components that are directly describing x_i, and the ones that are describing x_i's parts. For example, a variable that is describing a structural component of x_i, is also describing x_i. The definition introduced above is true because the semantic

relations were built to satisfy the axiom of generalized transitivity. This axiom is presented in Appendix A.

Some analysis activities may require access to certain specific mathematical components of a given model, like relationships and variables. These may be accessed indirectly by specializing on the mathematical components of x_i or directly if a class architecture is associated with them. In the latter instance, they can be obtained as follows:

$$\{\text{Variables describing } x_i\} = \{x_j/x_j \text{ is-describing } x_i \wedge x_j \text{ is-a-member-of VARIABLE-SUBCLASS} \wedge \text{VARIABLE-CLASS is-a GENERIC-VARIABLE}\}$$

$$\{\text{Relationships describing } x_i\} = \{x_j/x_j \text{ is-describing } x_i \wedge x_j \text{ is-a-member-of RELATION-SUBCLASS} \wedge \text{RELATION-SUBCLASS is-a RELATIONSHIP}\}$$

In the definitions presented above we have applied the property of generalized transitivity to the semantic relation is-describing and the property of transitivity to the semantic relation is-a-member-of. Likewise, we can identify more specific parts of the mathematical model, like:

$$\{\text{Rules describing } x_i\} = \{x_j/x_j \text{ is-describing } x_i \wedge x_j \text{ is-a-member-of RULE}\}$$

$$\{\text{Kinetic Equations describing } x_i\} = \{x_j/x_j \text{ is-describing } x_i \wedge x_j \text{ is-a-member-of KINETIC-EQUATION-SUBCLASS} \wedge \text{KINETIC-EQUATION-SUBCLASS is-a KINETIC-EQUATION}\}$$

$$\{\text{Concentration Variables describing } x_i\} = \{x_j/x_j \text{ is-describing } x_i \wedge x_j \text{ is-a-member-of Concentration}\}$$

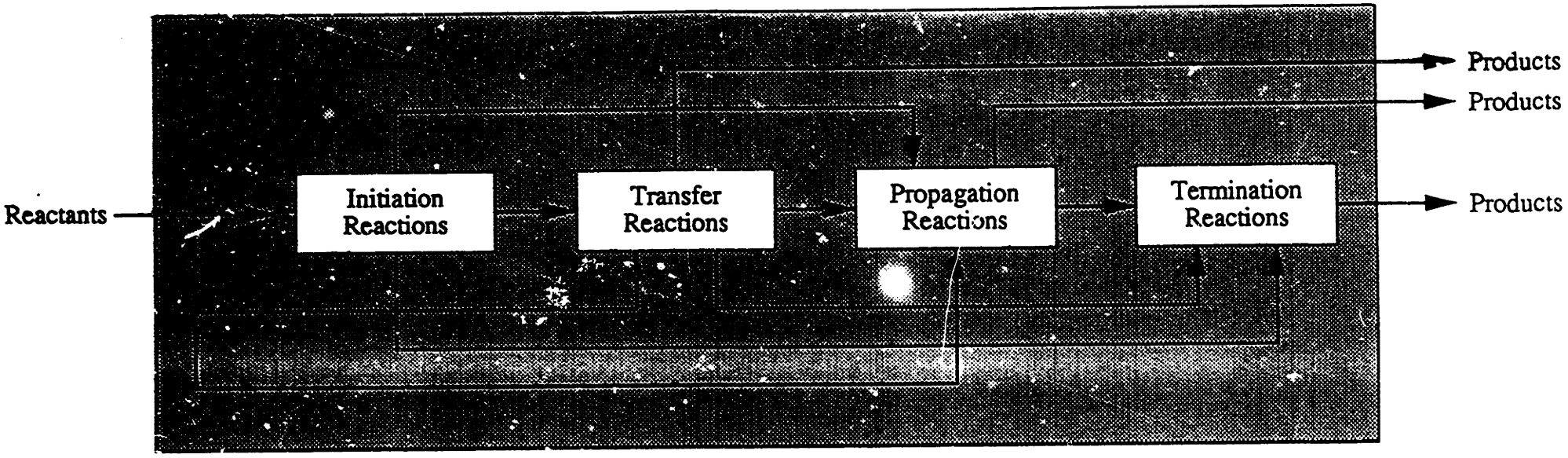
Once the particular components have been reached, their specific attributes can be accessed. Thus, variable units can be investigated, variables' values can be stored and retrieved, equations can be evaluated, antecedents of rules can be checked, etc. Also notice, that the ordering relations between semantic links necessitate that is-connected-by and is-composed-of links be searched when accessing all is-described-by links in a model. This is required because mathematical components may be found describing topological components at all levels in a model.

6.2.2 Accessing the topological components

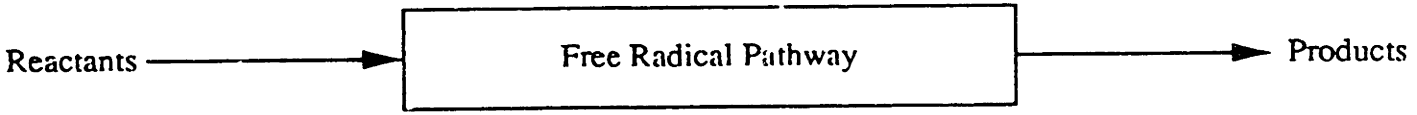
The methodology to access these components is similar to the one presented for mathematical elements and the definitions to be introduced here are also based on the axiom of generalized transitivity. Thus, the edges (e.g. bonds) associated with a structure x_i (e.g. chemical species) can be obtained as follows:

$$\{\text{Covalent Bonds attached to } x_i\} = \{x_j/x_j \text{ is-attached-to } x_i \wedge x_j \text{ is-a-member-of Covalent-Bond}\}$$

This operation gives us all the edges or ports of the structure (or metastructure): the ones that are directly attached to x_i and the edges that are attached to x_i's components. But, it may also be needed to access only certain edges of a structure. For example, Figure 6-1a shows the representation of a simple free radical pathway. During the course of a developmental activity we may want to represent it in a more abstract way, like the one depicted in Figure 6-1b. If we want to automate the abstraction process, we should have a mechanism to distinguish external from internal edges/ports. The external ports, the ones that are really connecting the object with its surroundings, can be identified by means of the following operation:



(a)



(b)

Figure 6-1: (a) A detailed representation of a free radical pathway
(b) An abstract representation of a free radical pathway

$$\{\text{External Edges/Ports attached to } x_i\} = \{x_j/x_j \in \text{External Edges/Ports attached to } x_i \wedge (\{\Gamma^-(x_j) \cap (\Gamma^+(x_i) \cup \{x_i\})\} \text{ is a singleton})\}$$

where $\Gamma^-(x_j)$ denotes ancestors of node x_j and $\Gamma^+(x_i)$ denotes descendants of node x_i. The explicitly structured models, generated by CRL using having well-defined semantics, allows the identification of very specific components of a model; and, makes possible access to partial views of the subgraphs that are comprising the digraph of a model (see chapter 5). Thus, it allows the CRL user to concentrate on "a specific view of a given model perspective". If, for example, the user wants to know the value of the variables that are describing the external ports of a structure x_i, it is first necessary to identify the relevant variable objects. They can be obtained as follows:

$$\{\text{Variables describing external ports of } x_i\} = \{x_j/x_j \text{ is-describing } y_k \wedge y_k \in \{\text{External Ports attached to } x_i\} \wedge x_j \text{ is-a-member-of GENERIC-VARIABLE}\}$$

After which, it is possible to query the value of the attribute "value" of the appropriate variable objects. For example, in analyzing pathway candidates for the thermochemical production of hydrogen from water, it is advantageous to optimize with respect to heat absorption and entropy change. Because CRL supports multiple viewing of the overall process, multivariable optimization is achievable without performing an a postorie search of the critical variables; CRL facilitates the identification and evaluation of prospective cycles by providing a unitary vehicle for encapsulating the multifarious knowledge required for system evaluation.

That in order to access particular components of a model we have not performed a brute force search is clear from the above. The structure of the model itself has helped us in "moving inside" a model in a much easier way than if it were just a plain digraph.

Similar procedures can be followed to identify components describing physicochemical phenomena, or structural aspects of the model, and will not be discussed.

Lastly, notice that if a CRL user wants to access topological or another type of connecting components, he/she will need to search along is-connected-by edges. However, the ordering relation introduced during the presentation of the generalized transitivity axiom indicates that the search should also be done along the is-composed-of links that are preceding the is-connected-by links. This guarantees the access to all the ports of a model, the ones directly attached to the model's root, and the ones attached to the other structural components.

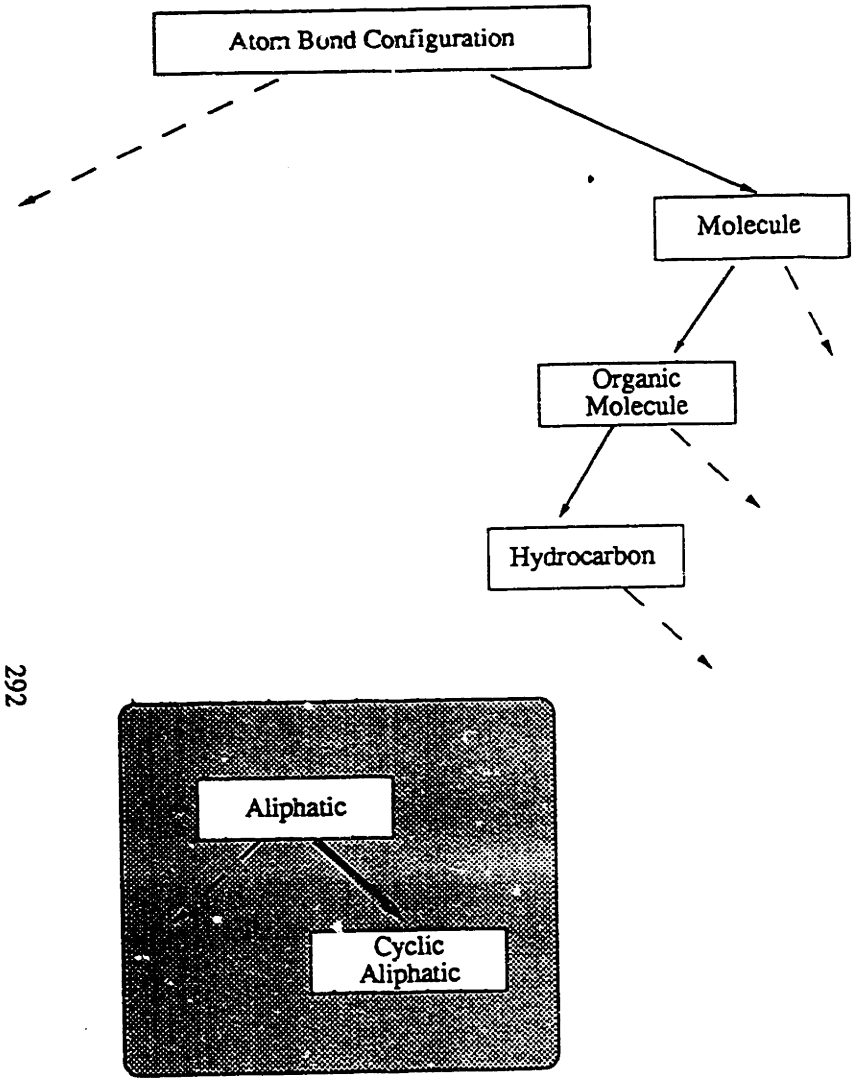
6.3 Formal Construction of Subclass Models

In chapter 5 we introduced the idea that a useful representation of chemical structure and chemical operations is composed of objects of different types; in preceding sections we described how the information contained in these objects is accessed; the burden is now to show how these components interact to achieve the objectives we seek. We will demonstrate how modeling elements are created and added to the various subclass models in the following sections. Herein classes are represented in **bold**; semantic relationships are represented in ***bold italic***.

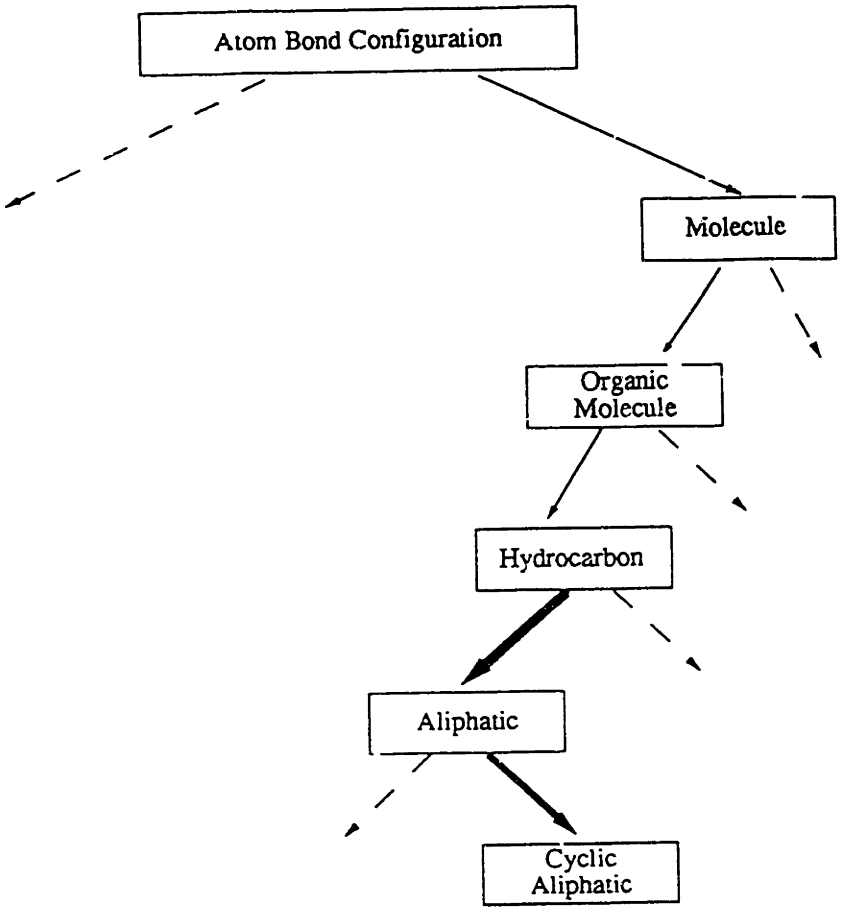
6.3.1 Atom-bond-configuration

Let us first consider extension of the abc class. Suppose it is advantageous to reason about cyclic aliphatic species directly. For this purpose, we create the modeling class cyclic-aliphatic and link it to the parent class aliphatic, thereby establishing set membership (see Fig. 6-2a).

cyclic-aliphatic ***is-a*** **aliphatic**



(a)



(b)

Figure 6-2: Model Class Addition: chemical structure

Since **aliphatic** is a specialization on the class **hydrocarbon**, it is advantageous to create a link between **aliphatic** and **hydrocarbon** (see Fig. 6-2b).

aliphatic *is-a* **hydrocarbon**

This allows the attributes describing **hydrocarbon** to be inherited to **aliphatic**; similarly, the link established between **aliphatic** and **cyclic-aliphatic** allows this information to be accessed from **cyclic-aliphatic** directly. Having created these links, any attribute or method residing in the class **aliphatic** or any of its super classes is inherited by **cyclic-aliphatic**. For example, if methods cyclic-p, compute-nonbonded-strain and compute-torsional-strain reside in **atom-bond-configuration** and methods compute-skeleton-chains and compute-longest-carbon-chain reside in **organic-molecule**, each of these methods as well as any additional descriptive attributes are directly accessible from the modeling class **cyclic-aliphatic**; no duplication of modeling effort is required. How methods and attributes are combined or mixed into the **cyclic-aliphatic** class is user controlled.

It may also be desirous to extend the utility of the **cyclic-aliphatic** class and its descriptors through class specific method and attribute addition (see Fig. 6-3). These are chosen to enhance the conceptual significance of the **cyclic-aliphatic** class. For example, the user may wish to include the following attributes in the class description:

Cyclic-aliphatic: attributes

identifier *(inherited from super class)*
old-identifier *(inherited from super class)*
name *(inherited from super class)*
parent-abc *(inherited from super class)*
progenitors *(inherited from super class)*
... *(inherited from super class)*
ring-strain
number-of-rings
ring-sizes
ring-number
torsional-strain
nonbonded-strain
bridged-atoms
...

Cyclic-aliphatic: methods

compute-identifier *(inherited from super class)*
compute-old-identifier *(inherited from super class)*
compute-parent-abc *(inherited from super class)*
compute-progenitors *(inherited from super class)*
... *(inherited from super class)*
compute-name (super class method modified)
compute-ring-strain
compute-number-of-rings
compute-ring-sizes
compute-ring-number
compute-torsional-strain
compute-nonbonded-strain
compute-bridged-atoms
monocyclic-ring-p
bicyclic-ring-p
fused-ring-p
...

Figure 6-3: Select attributes and methods of cyclic-aliphatic

ring-strain	*is-attribute-of*	**cyclic-aliphatic**
number-of-rings	*is-attribute-of*	**cyclic-aliphatic**
ring-sizes	*is-attribute-of*	**cyclic-aliphatic**
ring-number	*is-attribute-of*	**cyclic-aliphatic**
torsional-strain	*is-attribute-of*	**cyclic-aliphatic**
nonbonded-strain	*is-attribute-of*	**cyclic-aliphatic**
bridged-atoms	*is-attribute-of*	**cyclic-aliphatic**
...		

The value of these attributes are then established by the methods that have been associated with **cyclic-aliphatic**. Methods indicative of this class may include:

compute-ring-strain	*is-method-of*	**cyclic-aliphatic**
compute-number-of-rings	*is-method-of*	**cyclic-aliphatic**
compute-ring-sizes	*is-method-of*	**cyclic-aliphatic**
compute-ring-number	*is-method-of*	**cyclic-aliphatic**
compute-torsional-strain	*is-method-of*	**cyclic-aliphatic**
compute-nonbonded-strain	*is-method-of*	**cyclic-aliphatic**
compute-bridged-atoms	*is-method-of*	**cyclic-aliphatic**
monocyclic-ring-p	*is-method-of*	**cyclic-aliphatic**
bicyclic-ring-p	*is-method-of*	**cyclic-aliphatic**
fused-ring-p	*is-method-of*	**cyclic-aliphatic**
...		

In the above methods, those with the "compute-" prefix determine attribute values; those with the "-p" suffix are predicate methods returning boolean values. The remaining methods are invoked on instances of the class **cyclic-aliphatic** and are designed to raise the abstraction level (i.e. they allow the user to communicate using higher level

concepts). These methods may be called from outside the class **cyclic-aliphatic** provided they operate on instances of the **cyclic-aliphatic** class. The method bridged-atoms illustrates this concept; it calls bridged-atom-p, a predicate method of atom, from the class **cyclic-aliphatic**. Likewise, if reactivity assessment requires knowledge about ring structure, we can access that information by applying the semantic relation ***is-describing*** to an instance of **cyclic-aliphatic,** ICA, as shown below:

bridged-atoms-ICA *is-describing* (**atom-*1* ... atom-*n***)

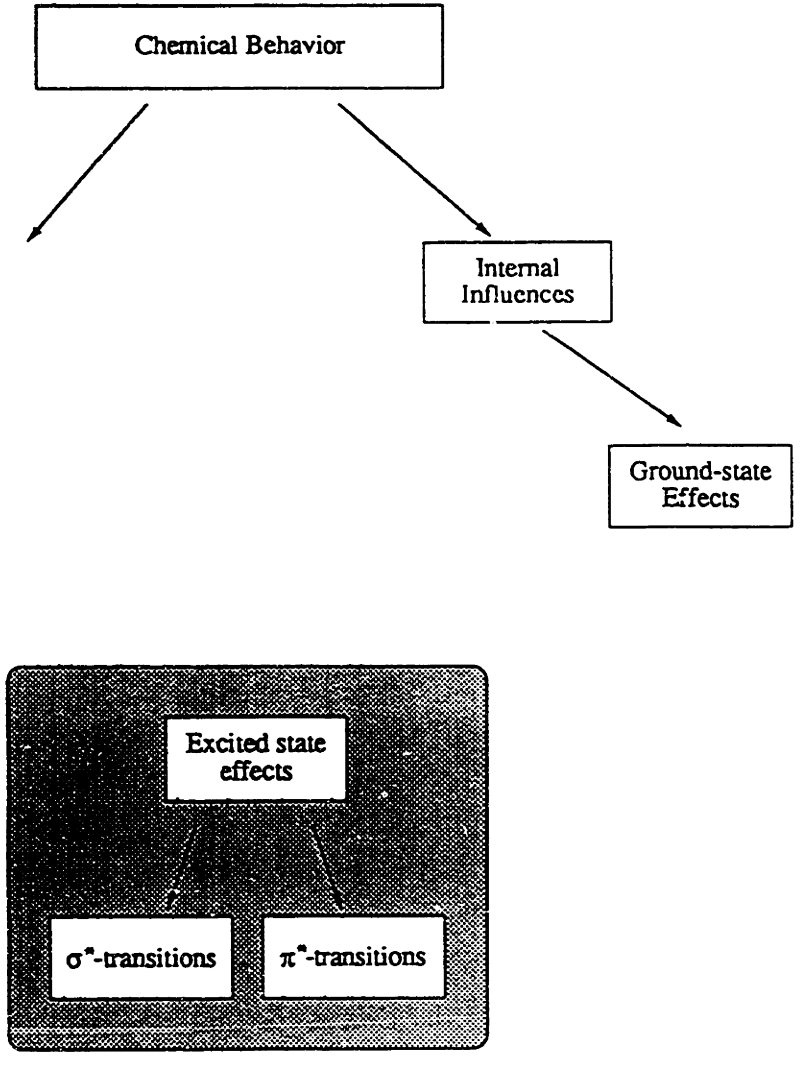
Notice also that although the methods compute-nonbonded-strain and compute-torsional-strain were inherited by **cyclic-aliphatic**, more sophisticated versions of these methods were supplied to **cyclic-aliphatic**. This would have been unnecessary if the original versions of these methods supported fully *all* cyclic configurations (i.e. a robust method given any ring configuration). We have found that ability to incrementally improve methods accelerates model development.

6.3.2 Chemical behavior

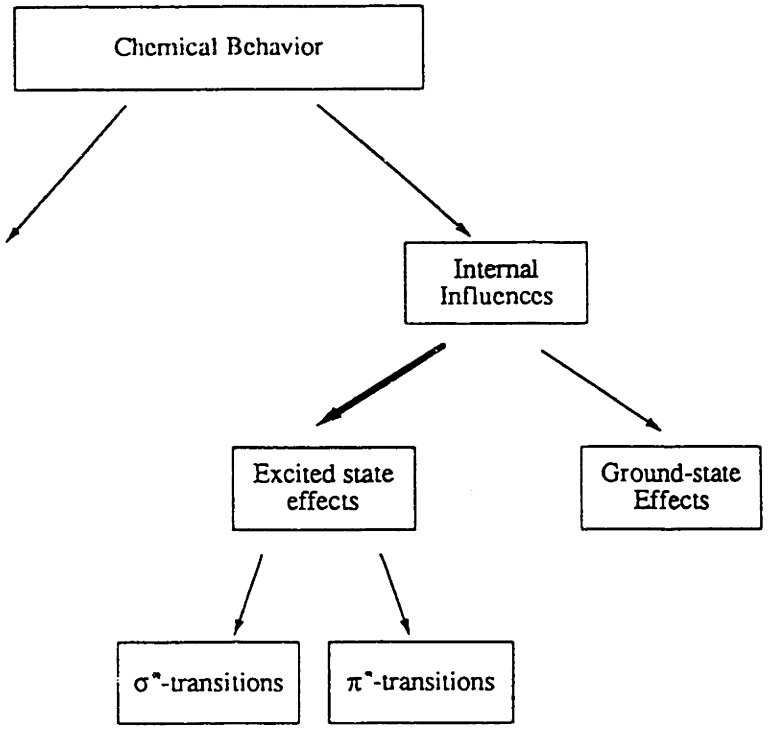
Suppose, next, we choose to embellish the modeling class **chemical-behavior** by developing the subclass **excited-state-effects**. We begin by developing a subclass architecture consistent with the existing organization: characterization of the electronic state. In this spirit, we have specialized **internal-influences** according to **ground-state-effects** and **excited-state-effects**. We do this using the semantic relation ***is-a*** (see Fig 6-4a):

excited-state-effects *is-a* **internal-influence**

Further, we specialize **excited-state-effects** according to the accompanying electronic transitions: transitions to σ^* and transitions to π^*; and link these subclasses to **excited-state-effects** as shown below (see Fig. 6-4b).



(a)



(b)

Figure 6-4: Model Class Addition: chemical behavior

σ^*-transitions	*is-a*	**excited-state-effects**
π^*-transitions	*is-a*	**excited-state-effects**

The addition of descriptive attributes and methods to each of these new modeling classes further characterize each class. To characterize **excited-state-effects** we add attributes and methods descriptive of singlet (S) and triplet (T) states. The states are refined further by energy level. For example ground states (S_0 ,T_0) are differentiated from their excited states (S_1 ,T_1) as well as from their higher states (S_2, T_2, and S_3, T_3). As before, we link these attributes using the the semantic relationship ***is-attribute-of***:

S_0	*is-attribute-of*	**excited-state-effects**
S_1	*is-attribute-of*	**excited-state-effects**
S_2	*is-attribute-of*	**excited-state-effects**
S_3	*is-attribute-of*	**excited-state-effects**
T_0	*is-attribute-of*	**excited-state-effects**
T_1	*is-attribute-of*	**excited-state-effects**
T_2	*is-attribute-of*	**excited-state-effects**
T_3	*is-attribute-of*	**excited-state-effects**
...		

Additional attributes may also be added. The degree to which it is advantageous to add additional class descriptors (eg. attributes, methods) is defined by the scope of the modeling efforts.

Methods characterizing **excited-state-effects** are linked to their associated class using the semantic relation ***is-method-of***. These methods can be grouped according to whether the result is used by the attribute's compute method:

compute-S_0	*is-method-of*	**excited-state-effects**
compute-S_1	*is-method-of*	**excited-state-effects**
compute-S_2	*is-method-of*	**excited-state-effects**
compute-S_3	*is-method-of*	**excited-state-effects**
compute-T_0	*is-method-of*	**excited-state-effects**
compute-T_1	*is-method-of*	**excited-state-effects**
compute-T_2	*is-method-of*	**excited-state-effects**
compute-T_3	*is-method-of*	**excited-state-effects**
...		

or by methods used to derive additional properties of the attribute space. The latter group may include generic methods such as assess-energy-gap, which computes the energy difference between two energy states(eg. S_1 and S_2 or S_1 and T_1); or predicate functions which access feasibility. Several of these are shown below; their function is transparent:

assess-energy-gap	*is-method-of*	**excited-state-effects**
allowed-transition-p	*is-method-of*	**excited-state-effects**
forbidden-transition-p	*is-method-of*	**excited-state-effects**
fluorescence-p	*is-method-of*	**excited-state-effects**
phosphorescence-p	*is-method-of*	**excited-state-effects**
internal-conversion-p	*is-method-of*	**excited-state-effects**
internal-system-crossing-p	*is-method-of*	**excited-state-effects**
...		

Together these elements give the modeling class utility. Notice that the capability of moving electrons between energy levels requires that **atom** or **bond** have attributes that describe atomic and molecular orbitals. Given these attributes, the movement of an

electron from one energy level to another can be orchestrated by methods residing in **atom**, **bond**, and **ab-initio-operator** (e.g. **ionization** or **single-electron-transfer**). In the latter case, a well defined abstraction barrier is established which helps maintain program modularity. Understandly, if the modeling effort does not require analysis of electron movement the functionality need not be included. If it is required, CRL has the capability to expand and accommodate the modeling effort regardless of scope. A state description of **excited-state-effects**, after attribute and method addition, is shown in Figure 6-5.

Similarly, the subclasses of **excited-state-effects**, **σ^*-transitions** and **π^*-transitions**, are described by the transitions which characterize them. These include σ to σ^*, *n* to σ^*, *n* to π^*, and π to π^* transitions. The corresponding attribute and methods are linked to their respective classes as described earlier. Very briefly, they are shown below:

$\sigma \rightarrow \sigma^*$	*is-attribute-of*	**σ^*-transitions**
$n \rightarrow \sigma^*$	*is-attribute-of*	**σ^*-transitions**
...		
$\pi \rightarrow \pi^*$	*is-attribute-of*	**π^*-transitions**
$n \rightarrow \pi^*$	*is-attribute-of*	**π^*-transitions**
...		

and,

compute-$\sigma \rightarrow \sigma^*$-transition	*is-method-of*	**σ^*-transitions**
compute-$n \rightarrow \sigma^*$-transition	*is-method-of*	**σ^*-transitions**
...		
compute-$\pi \rightarrow \pi^*$-transition	*is-method-of*	**π^*-transitions**
compute-$n \rightarrow \pi^*$-transition	*is-method-of*	**π^*-transitions**
...		

Excited-state-effects: attributes

identifier *(inherited from super class)*
old-identifier *(inherited from super class)*
bonding-orbital *(inherited from super class)*
anti-bonding-orbital *(inherited from super class)*
dominate-behavior*(inherited from super class)*
competitive-behavior*(inherited from super class)*
... *(inherited from super class)*
S_0
S_1
S_2
S_3
T_0
T_1
T_2
...

Excited-state-effects: methods

compute-identifier *(inherited from super class)*
compute-old-identifier *(inherited from super class)*
compute-bonding-orbital *(inherited from super class)*
compute-anti-bonding-orbital *(inherited from super class)*
compute-dominate-behavior *(inherited from super class)*
compute-competitive-behavior *(inherited from super class)*
... *(inherited from super class)*
compute-S_0
compute-S_1
compute-S_2
compute-S_3
compute-T_0
compute-T_1
compute-T_2
compute-T_3
assess-energy-gap
allowed-transition-p
forbidden-transition-p
fluorescence-p
phosphorescence-p
internal-conversion-p
internal-system-crossing-p
...

Figure 6-5: Select attributes and methods of excited-state-effects

We can also specialize a modeling class by characterizing several of its dominate or critical properties. For select operations, such as composite or ab initio operations, specialization can further refine the search space and afford greater (search) efficiency. The semantic relationship ***is-characterized-as*** is used for this purpose. It provides a window for viewing the more interesting behaviors/structures associated with a class. For example,

excited-state-transition	***is-characterized-as***	$\pi \rightarrow \pi^*$
excited-state-transition	***is-characterized-as***	$n \rightarrow \pi^*$

or,

electronic -state	***is-characterized-as***	excimer
electronic-state	***is-characterized-as***	exciplex

Using this semantic relation, the user may investigate alternative pathways by specifying a different characterization. Consistency is maintained by the multiple-context modeling utility of CRL. Moreover, the context dependent functionality of the chemical behavior modeling element (see chapter 5) and the dynamic association of a behavior with a species allows complete elucidation of reaction alternatives in a multifaceted manner: alternatives may be investigated independently with respect to structural character, electronic state, reaction environment, mechanism, etc. For example, once a semantic pathway is generated, the user can assess the effect of changing pathway inputs: reaction temperature, reaction medium (i.e. solvent), resonance state, reaction site, etc. This functionality allows what-if type questions to be resolve in real-time. Section 6.4 illustrates these points.

6.3.3 Ab initio operator

Previously we have described the generation of class and subclass models with a focus on model class addition. Our attention now turns to the operators: methods that perform the

actual transformation. Bond formation, bond cleavage, electron excitation, and electron decay are a few such examples. These operators lie at the heart of CRL.

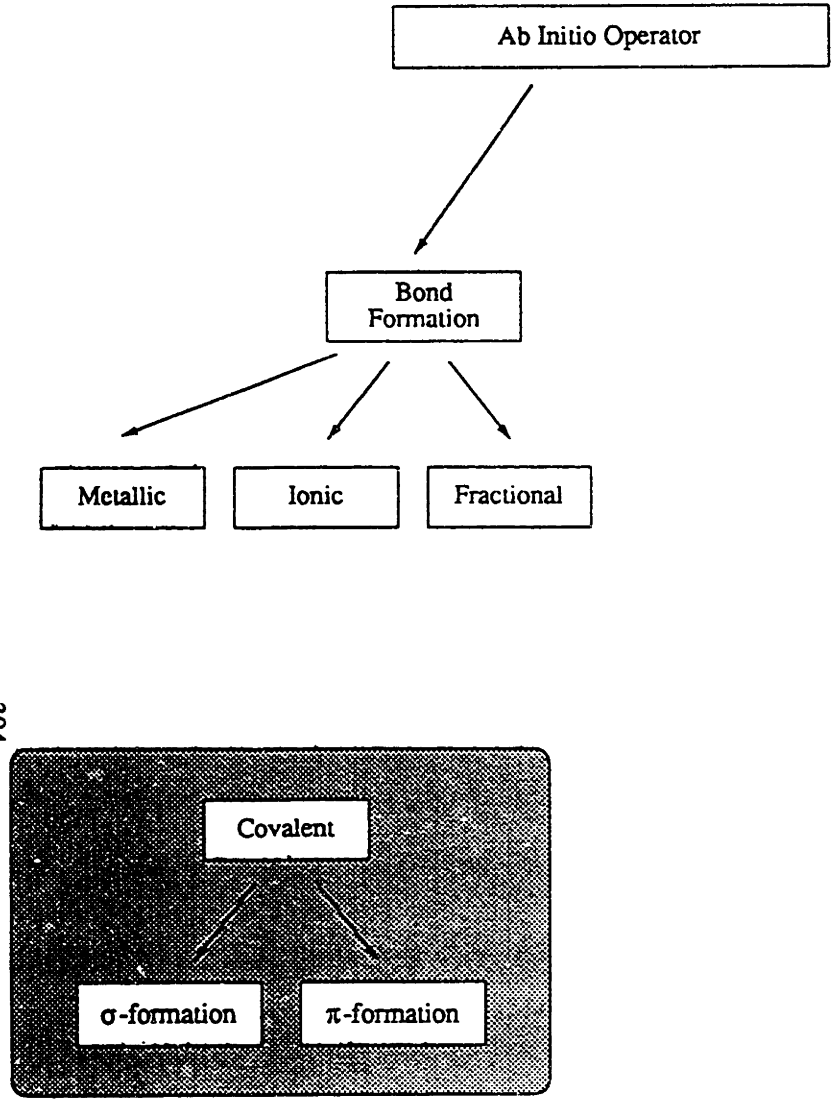
We will illustrate the development of these operators by creating the ab initio modeling element, **covalent-bond**. We will establish in this development: (1) the contract between **covalent-bond** and **ab initio operator, K_{ai}** (so that its input and output are recognizable); (2) the criteria on enabling conditions; (3) how criteria can be relaxed, tightened, augmented, or supplemented; (4) how additional functionality can be added to **covalent-bond** (reflecting advances in our understanding of the system).

We initiate the development of **covalent-bond** by creating the modeling element and establishing the normal superclass links using the semantic relationship ***is-a*** (see Fig. 6-6):

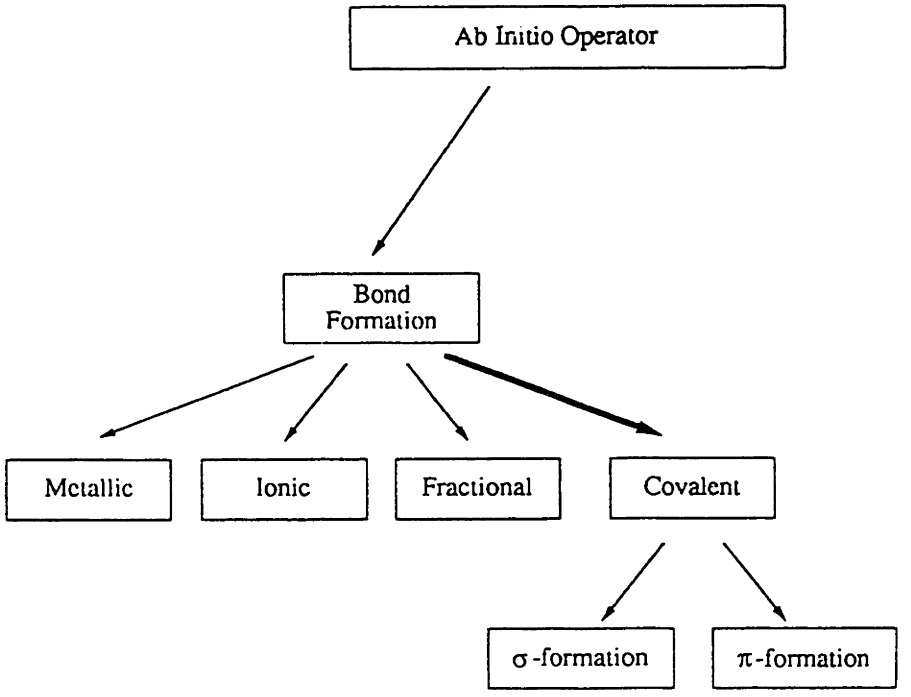
covalent-bond	***is-a***	**ab initio operator**
σ-formation	***is-a***	**covalent-bond**
π-formation	***is-a***	**covalent-bond**

However, we will not specify the attributes of **covalent-bond**, **σ-formation**, or **π-formation** as we have done previously: these are used for the purposes of bookkeeping; how the attributes are chosen and ultimately used depends on the generality and robustness of the supporting encoded methods.

To affect a transformation **covalent-bond** requires methods. In particular, we need to establish generic or dominant class methods: methods responsible for primary transformations. The dominant method of **covalent-bond** is make-covalent-bond. Make-covalent-bond establishes a **covalent-bond** between two reaction centers when enabling conditions are achieved. It also performs all the necessary bookkeeping to



(a)



(b)

Figure 6-6: Model Class Addition: ab initio operator

ensure the bond is recognizable by the abc. Expressed in pseudocode convention [14], these methods take the form:

```
input
initialize
if  feasible-p
    then enable-transformation
    return
end
return
```

where enable-transformation performs the transformation and feasible-p, a set of one or more predicate methods, represents the enabling conditions, <K_{ai}-enabling-cond>, of the transformation .

The syntax of K_{ai}, given in chapter 5, is shown below in Backus-Naur-Form (BNF):

<ab-initio-operator> ::= {[<K_{ai}>]$^+$}

<K_{ai}-operator> ::= Bond formation | Bond Cleavage | Ionization | Single Electron Transfer | Electron Excitation | Electron Decay

<K_{ai}> ::= <K_{ai}-operator><K_{ai}-input><K_{ai}-enabling-cond><output>

<K_{ai}-input> ::= {[<reaction-center>]$^+$}

<reaction-center> ::= {[<atom>]$^+$}

<K_{ai}-enabling-cond> ::= {[<K_{ai}-condition>]$^+$}

<K_{ai}-condition> ::= {[physicochemical-requirements]$^+$}

<physical-requirements> ::= (IS-ABSTRACTING <physicochemical-requirements>)

The syntax of **ab-initio-operator** identifies the following elements of <K_{ai}>: <K_{ai}-operator>, <K_{ai}-input>, <K_{ai}-enabling-cond>, and <output>. Notice that no restriction

is placed on where these elements reside; their positioning is implementation dependent. [In our implementation of CRL, we have chosen to associate <K_{ai}-enabling-cond> with the highest abstraction level possible. This helps maintain program modularity.] Continuing with our task, we recognize that **covalent-bond** requires the specification of <K_{ai}-enabling-cond>, <K_{ai}-input>, and <output>. We will focus on <output> and <K_{ai}-enabling-cond> since the specification of <K_{ai}-input> is transparent (i.e. a list of one or more reaction centers).

The <output> of **covalent-bond** is an object that represents a covalent bond between two reaction centers. This object (i.e. instance of **covalent-bond**) is not a single (independent) entity. It knows membership and has pointers that connect it to the new **abc,** of which it is part. The new **abc** recognizes it as a covalent bond. The probability of a non-nil <output> being returned is dependent on the properties of the reaction centers and level of sophistication encoded into <K_{ai}-enabling-cond>.

Specific criteria must be satisfied for a covalent bond to be formed between two atoms. These criteria are based on the physical laws of nature and tempered by our knowledge of the bonding system. They are encoded into the procedure <K_{ai}-enabling-cond> using methods.

The general algorithm for <K_{ai}-enabling-cond> is:

```
<Kai-enabling-cond>:
        input
        initialize
        if  feasible-p
            then true
            return
        end
        return
```

where feasible-p represents a list of predicates procedures specifying the criteria for feasibility. The pseudocode procedural form of <K_{ai}-enabling-cond> for the modeling class, **covalent-bond**, is given below:

<covalent-bond-enabling-cond>
input: reaction-center-pair
initialize
reaction-center-1 ← first element of reaction-center-pair
reaction-center-2 ← last element of reaction-center-pair
if favorable-interatomic-distance-p **and**
potentially-stable-bond-formation-p **and**
available-bonding-electron-p **and**
proper-orbital-symmetry-p **and**
conservation-laws-upheld-p
then true
return
end
return

The feasibility criteria, following the "if" construct, are implemented as predicates methods. They perform the following evaluations: favorable-interatomic-distance-p assesses the interatomic distances between the two reaction centers to determine whether bond formation is likely; potentially-stable-bond-formation-p verifies that the steric repulsion between the reaction centers is less than the potential energy of the system and that the entropic term for bond formation is less than the enthalpic term for bond formation; available-bonding-electron-p checks the availability of bonding electrons; proper-orbital-symmetry-p determines whether the bonding orbitals are in phase; conservation-law-upheld-p validates whether mass, energy, and momentum have been conserved. These methods are associated with **covalent-bond** using the semantic relationship ***is-method-of***, as we have shown previously:

favorable-interatomic-distance-p ***is-method-of*** **covalent-bond**

potentially-stable-bond-formation-p *is-method-of* **covalent-bond**

available-bonding-electron-p *is-method-of* **covalent-bond**

proper-orbital-symmetry-p *is-method-of* **covalent-bond**

Notice that conservation-laws-upheld-p is not associated with **covalent-bond**. This method(s) is associated with $\mathbf{K_{ai}}$ allowing it to be accessed by all subclass modeling elements of $\mathbf{K_{ai}}$.

We can also specialize these methods according to physical requirements and chemical requirements. This provides an abstraction barrier between what is true (i.e. the physical laws) and what is believed to be true (i.e. current theory). For example, suppose that favorable-interatomic-distance-p and potentially-stable-bond-formation-p are physical requirements while available-bonding-electron-p and proper-orbital-symmetry-p are chemical requirements. We represent this to **covalent-bond** by associating the top level methods (i.e. methods associated with $\mathbf{K_{ai}}$), chemical requirements and physical requirements, to **covalent-bond** directly (i.e. we override the method inherited to the model class). This is accomplished using the semantic relationship *is-method-of*, as was demonstrated earlier for cyclic-p (see Section 6.3.1)

physical-requirement *is-method-of* **covalent-bond**

chemical-requirement *is-method-of* **covalent-bond**

We then tailor these methods by associating the methods of **covalent-bond** to them. This is achieved using the semantic relationship *is-composed-of*:

physical-requirment *is-composed-of* (favorable-interatomic-distance-p, potentially-stable-bond-formation-p)

chemical-requirement *is-composed-of* (available-bonding-electron-p,
proper-orbital-symmetry-p)

We have chosen the above classification because CRL, being written on top Lisp, accepts keyword arguments [15]. Keyword arguments give the user the ability to tailor (e.g. relax, tighten, augment, or supplement) the requirements for covalent bond formation by passing in values that add, delete, or modify the feasibility criteria. As an illustration, suppose the generic method make-covalent-bond has keyword arguments (i.e. arguments with a colon prefix) complete-requirements, partial-requirements, and modify-requirements as shown below:

(make-covalent-bond arguments *:complete-requirements*
:partial-requirements
:augment-requirements)

Now by passing in values for the various keywords, we can modify how the feasibility criteria <covalent-bond-enable-cond> is used for a single evaluation or a series of evaluations. For example,

(make-covalent-bond **reaction-center-pair** *:complete* true)

would return a covalent bond, provided all criteria of <covalent-bond-enable-cond> were met while,

(make-covalent-bond **reaction-center-pair**
:partial physical-requirements)

would return a covalent bond provided the criteria associated physical requirements were met. The feasibility criteria of <covalent-bond-enable-cond> can also be augmented using the keyword augment. In this case a function is passed in directly:

(make-covalent-bond **reaction-center-pair** *:augment* spin-coupling-p)

Having established the methods of **covalent-bond**, the choice of attributes now becomes apparent: they are the elements of these methods. Such selection will accelerate method development and method modification.

6.3.4 Composite operator

Suppose now we wish to add an additional operator to the model class composite operator (**K**). For example, consider the addition of radical-disproportionation; a transformation that depicts the following:

$$R_1 + R_2 \rightarrow R'_1 + R''_2$$

or more specifically,

$$^{\bullet}C_2H_5 + {}^{\bullet}C_3H_5 \rightarrow C_2H_4 + C_3H_8$$

$$\rightarrow C_2H_6 + C_3H_8$$

As before, the addition of $\mathbf{K_{radical\text{-}disproportionation}}$ to the model class requires that links be established between composite-operator and bimolecular-operator, bimolecular-operator and radical-operator, and, radical-operator and radical-disproportionation:

bimolecular-operator	*is-a*	**composite-operator**
radical-operator	*is-a*	**bimolecular-operator**
radical-disproportionation	*is-a*	**radical-operator**

Like $\mathbf{K_{ai}}$, the modeling element **K**, performs transformations. This necessitates that procedures be developed and associated in **K**. Unlike $\mathbf{K_{ai}}$, **K** procedures are for known transformations; not theoretically feasible transformations (see chapter 5).

To identify the procedures required of $\mathbf{K}_{\textbf{radical-disproportionation}}$ we return to the syntax presented in chapter 5. It is shown below:

<composite-operator> ::= <inputs><enabling-conditions><ab-initio-operator>
<output> [<composite-operator>]

<input> ::= <react-cond><abc><cb>.

<output> ::= <reaction>.

<abc> ::= {[(Instance-of abc)]$^{+}$}.

<cb> ::= {[(Instance-of cb)]$^{+}$}.

<reaction-cond> ::= {(Instance-of reaction-conditions)}.

<enabling-conditions> ::= {[<conditionals>]$^{+}$}.

<conditionals> ::= [[[(Instance-of user preferences)]$^{+}$
{[(Instance-of physical requirements)]$^{+}$}]].

The syntax of **K** identifies the elements of any subclass modeling element (e.g. $\mathbf{K}_{\textbf{radical-disproportionation}}$). They are: <inputs> <enabling-conditions> <ab-initio-operator> <output>. Each of these decompose as shown above. [The decomposition of <ab-initio-operator> is given in the previous section.]

Additionally, we review the definition of **K** to identify the procedures that must be provided to **K**.

definition: A composite operator, **K**, is composed of K_f, $K_{get\text{-}sites}$, and K_t. **K** identifies potential reaction sites by calling $K_{get\text{-}sites}$ on the structure(s). K_t is the applied to those sites forming a new set of **abc's** when K_f, a feasibility predicate, returns true.

We now direct our attention at the development of these procedures: K_f, K_t, and $K_{get\text{-}sites}$. In the context of radical disporportionation these procedures will evaluate the following:

1) K_f evaluates the feasibility of radical disproportionation (e.g. R_1 and R_2 are both radicals, energy in the reaction environment is less than that required to cleave the newly formed bond, relative stability of products, etc.)

2) $K_{get\text{-}sites}$ identifies potential reaction sites in R_1 and R_2.

3) K_t supplies a sequence of ab initio operators and applies these operators to the reaction sites identified by $K_{get\text{-}sites}$ to obtain the desired transformation.

The feasibility predicate, K_f, for $K_{radical\text{-}disproportionation}$ is similar in spirit to <K_{ai}-enabling-cond> with the exception that the user can encode personal preferences. For example, suppose for safety reasons that the user's interest is focused on a substituted product. This preference could be encoded into K_f by passing in a predicate test, substituted-p, that would be applied to reaction-centers.

K_f for radical-disproportionation is shown below using the notation presented in Section 6.3.3:

K_f:

input: substrate-list
when substrates are radicals
collect radical centers into potential-reaction-centers
evaluate potential-bonds between potential-reaction-centers
for each bond in potential-bonds evaluate bond-energy
when bond is stable in reaction-environment
and user-preferences satisfied

```
                                    collect potential-reaction-center-pair
                                        into potential-reaction-sites
                                    return potential-reaction-sites
                                return
                            return
                        end
                        return
```

Notice the usage of inputs (i.e. <abc>, <cb>, and <reaction-condition>) in K_f: substrates and substrate-list are sets of <abc>'s; radicals are identified using <cb>; bond stability is assessed in the context of <reaction-condition> (i.e. the **reaction-environment**). Additionally note that multiple products may be found since there may be several favorable sites; potential-reaction-sites are collected on the basis of thermodynamic stability relative to the reaction-environment.

To enhance the performance of CRL we have employed a three pass filter system: the first filter is the object system itself (as described in chapter 5); the second filter assesses the thermodynamic stability of the new bond(s) relative to the old bond(s); the third filter assesses the thermodynamic state of the products versus the reactants in the reaction environment. The reader will note that the second pass filter is embedded in K_f (i.e. when bond is stable ...) while the third pass filter is embedded in $K_{get\text{-}sites}$ (i.e. stability-acceptable-p). The embedding of the final filter into $K_{get\text{-}sites}$ is necessitated because K_t must be applied (i.e. the products must be generated) before the final filter can evaluate the systems in the context of the environment. To avoid unnecessary application of K_t the first and second pass filter systems should be as selective as possible

$K_{get\text{-}sites}$ is responsible for producing products given a set of potential reaction sites. It applies K_t to potential reaction center combinations and assesses the relative stabilities of the products (i.e. third pass filter). The procedure for $K_{get\text{-}sites}$ is given below:

$K_{get\text{-}sites}$:

input: potential-reaction-sites

for each reaction-center-pair in potential-reaction-sites

apply K_t to reaction-center-pair

collect potential-structure-pair into potential-structures

return potential-structures

sort potential-structure-pair by stability

when stability-acceptable-p of potential-structure-pair

collect potential-structure-pair into product-pairs

return product-pairs

end

return

The call to K_t enables the desired transformation. This transformation (encoded below in K_t) begins by cleaving the bond connecting a mobile moiety (e.g. H) to a parent atom adjacent to reaction-center-1. Products are then produced by creating new bonds between these centers using generic methods (e.g. make-covalent-bond and cleave-bond) to achieve bond formation and bond cleavage. K_t uses these generic methods to access knowledge contained in K_{ai}.

The procedure for K_t is:

K_t:

input: reaction-center-pair

initialize reaction-center-1 ← first element of reaction-center-pair

reaction-center-2 ← second element of reaction-center-pair

cleave-bond mobile moiety adjacent to reaction-center-1

biradical ← reaction-center-1 structure

product-1 ← apply make-covalent-bond to mobile moiety and
reaction-center-2

product-2 ← apply make-covalent-bond to biradical

end

return

By changing the transformation code and by adding selector functions we can enhance K_t to enable additional disproportionations. These may include transformations depicting disproportionation of such radicals as $R_3C\dot{C}O_2$ and $R_2HC\dot{C}O_2$.

Finally, we associate the procedures K_f, K_t, and $K_{get\text{-}sites}$ to **K_{radical-disproportionation}** using the semantic relation *is-method-of*:

K_f ***is-method-of*** **radical-disproportionation**

$K_{get\text{-}sites}$ ***is-method-of*** **radical-disproportionation**

K_t ***is-method-of*** **radical-disproportionation**

We also establish a second (redundant) link using the semantic relationship ***is-described-by***:

radical-disproportionation ***is-described-by*** K_t

radical-disproportionation ***is-described-by*** $K_{get\text{-}sites}$

radical-disproportionation ***is-described-by*** K_f

This link provides an additional means of accessing these procedures in a way that allows us to chain upon them in the synthetic and retrosynthetic direction.

Now **Kradical-disproportionation** is fully recognized an useable by **K**. An evaluation of **Kradical-disproportionation** on R_1 and R_2 will return a **reaction** or set of **reactions** (reflecting multiple favored products). Each **reaction** constructed will contain values for the attributes (e.g. reactants, products, stoichiometry, reaction-environment, enabling conditions, etc.) as specified by the modeling element **reaction** (see Section 6.4.1).

6.3.5 Graph theoretics of the model-class structure

In typical algorithmic computer languages a compound algorithm is built from simpler ones by rules of temporal sequencing and data flow. Algorithms can be temporarily concatenated (e.g. concatenated), selected among (e.g. conditionals) and iterated (e.g. do loops). Data flow is indicated explicitly by functional composition and implicitly by side effects on shared variables. In CRL, a composite object representing a model class is built from simpler ones by linking some of their parts. The method of connection is the declaration of specification semantic relations among the parts.

The problem of formally explaining the constructs that are needed to create a model class is made up of the two subproblems of syntactic and semantic description. The specific written forms allowed by a language constitute its syntax; the actions involved by a form are expressed through its semantics. Although the set of legal sentences of a language can in theory be infinite, the grammatical formalism used to define must necessarily be finite. Chomsky [13] expeditiously mapped the finite to the infinite using the notion of production rules. Much of the same formalism, but in slightly different notation, is present in the Backus-Naur-Form (BNF) grammars; chapter 5 presented a description of CRL's syntactic structures employing a compact BNF grammar.

A detailed analysis of the model class definition (<class>) introduced in chapter 5 and illustrated in the previous sections, indicates that the creation of a model class starts with the model root and continues in a top-down fashion. This concept is recursively applied to each one of the components of the class being defined. The examples discussed previously, supported by thee definitions and syntax of CRL, define this process to be modular regardless of the mechanism used for their formation (i.e. class combination or new class creation). The gradual definition of a composite object allows the decomposition digraph to be associated with it at any stage of its development.

A digraph (or directed graph) G(X , U) consists of two sets:

i) a finite set $X=\{x_1 , x_2 , \ldots , x_n\}$, whose elements are called *nodes.*

ii) a subset U of the Cartesian product $X \times X$, the elements of which are called edges.

For an edge (x_i , x_j) the node x_i is the *initial end-point* and the node x_j is the *final end-point.* In G(X , U) the following definitions will be used:

•A node x_j is called *successor* of x_i if $(x_i , x_j) \in U$; the set of all successors of x_i is denoted by $\Gamma^+(x_i)$.

•A node x_i is called *predecessor* of x_j if $(x_i , x_j) \in U$; the set of all predecessors of x_j is denoted by $\Gamma^-(x_j)$.

•Any node x_i such that there exists a path from x_j to x_i is called *descendant* of x_j. We shall denote the set of descendants of a node x_j by $\hat{\Gamma}^+(x_j)$.

•Any node x_j such that there exists a path from x_j to x_i is called *ancestor* of x_i. We shall denote the set of ancestors of a node x_i by $\hat{\Gamma}^-(x_i)$.

Each digraph G=(X , U) determines a binary relation, R_G, on the set X of its nodes, through the rule

$$x_i \; R_G \; x_j \Leftrightarrow (x_i , x_j) \in U$$

Using the defined terminology, R_G is the relation "is a predecessor of" on the node-set X. Conversely, each binary relation R on a set X determines a digraph $G=(X , U_R)$, where

$$U_R = \{(x_i , x_j) \in X \times X / x_i \; R \; x_j \}$$

that is called the digraph of R.

The digraph associated with a model class is referred to as the *Model Class Decomposition Digraph* (MCDD). The MCDD represents the net result of the model class decomposition taking into account all the components created at that time. The basic idea is simple and is schematically depicted in Figure 6-7. At stage 1, the model root A has only one constituent entity called B. At stage 2, the entity C has been incorporated into the sub-structure of B; but, C is not a simple entity, it is a composite object itself (e.g. a covalent bond is-described-by a set of mathematical components). Consequently, its whole structure is pasted into B's substructure. Stage 3 shows that B's substructure has been further specified by incorporating an additional composite object called D. Following with B's definition, stages 4-6 show that a new constitutive part, named E, is being gradually created. This presents a contrast with the previous incorporation of entities C and D: they had pre-defined sub-structures. This process will continue until A's structure has been completely specified.

The modularity of the model class definition process allows the modeler to make use of pre-existing classes or to gradually define new ones. This feature combined with the communicative properties of the semantic relations make possible the modification or upgrading of a model class with the minimum of effort.

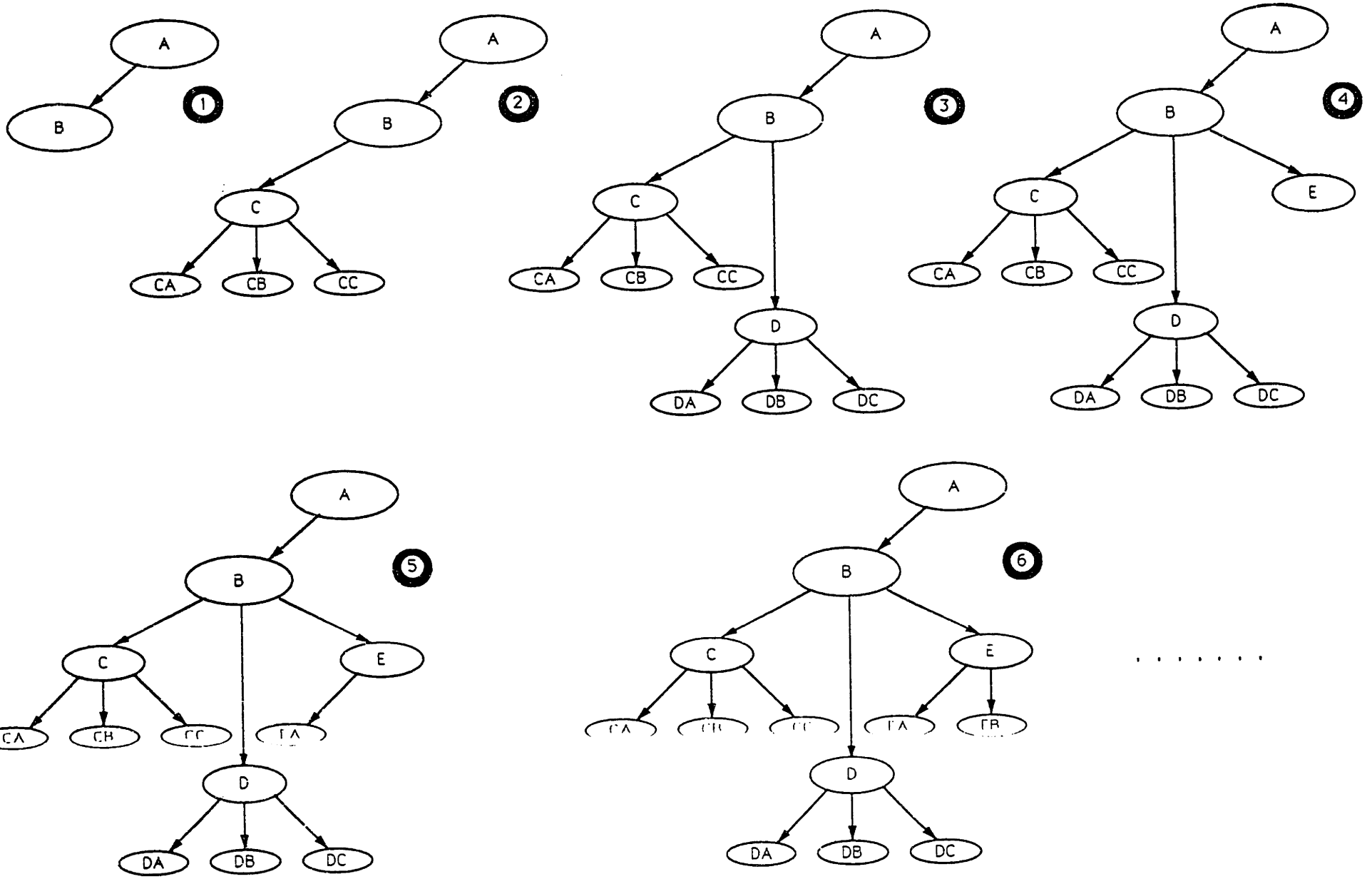


Figure 6-7: Schematic illustration of the generation process of a MCDD

MCDDs are made up of nodes and edges. The digraph's *nodes* have the particularity of being classes; each one of them corresponds to a constituent entity of the composite object that the model class is representing at that time. Pairs of nodes in a digraph are linked by *edges*. In a MCDD edges describe the semantic relations that victualed the entities represented by the nodes. Any edge of a MCDD can be associated with any of the following specification semantic relations: ***is-composed-of***, ***is-connected-by***, and ***is-described-by***. For example,

A MCDD is a digraph $G=(X , U_R)$, where:

(a) $X = \{x_1 , x_2 , \ldots , x_n\}$ is the set of nodes representing classes of modeling objects.

(b) $U_R = \{ (x_i, x_j) \in X \times X / x_i R x_j \}$ is the set of edges connecting nodes x_i and x_j through the three specification oriented semantic relationships, mentioned above.

A MCDD can be characterized as an acyclic digraph whose starting node, S, has a nil predecessor (i.e. $\Gamma^-(S) = 0$) is labelled with the name of the system the model is intended to represent. The successor nodes of S, $\Gamma^+(S)$, are labelled with the names of the first level components of the model. Each of these first level constituent entities may itself be decomposed into second level entities, which will be represented by new successor nodes in the diagraph (i.e. $\Gamma^+(\Gamma^+(S))$. Thus, the model of any node, x_i, is the subgraph of the MCDD generated by x_i and its descendants of $(\hat{\Gamma}^+(x_i) \cup \{x_i\})$, that is

$$MCDD(x_i) = (\{\hat{\Gamma}^+_{MCDD}(x_i) \cup \{x_i\}\}, U_R(x_i))$$

where $\hat{\Gamma}^+_{MCDD}(x_i)$ is the set of descendants of x_i in MCDD and $U_R(x_i)$ is given by

$$U_R(x_i) = \{(x_a , x_b) \in \{ \hat{\Gamma}^+_{MCDD}(x_i) \cup \{x_i\} \} \times \{\hat{\Gamma}^+_{MCDD}(x_i) \cup \{x_i\}\} / x_a R x_b\}$$

Formally, a MCDD represents a hierarchical decomposition of a model into constituent entities. Since this decomposition must ultimately terminate, a MCDD should not have directed circuits. Thus, MCDDs are digraphs that have the following properties:

Strict Hierarchy: No component node can appear in the path that has as initial end-point one of its constituent entity nodes. Ergo, no component can have a decomposition which eventually contains a constituent entity of the same type.

Uniformity: Any two nodes with the same label have attached the same constituent entities. This means that any two components of the same type have the same isomorphic decomposition structures. (Isomorphic here means copies of one another.)

These properties are particularly important because they guarantee the development of a reasonable model and easy access to its different views, components, etc. An additional property, resulting from the uniformity axiom is that all MCDDs have the capability to specify the digraphs that are associated with the instances of the model class. The instantiation of a MCDD generates a *Model Decomposition Digraph* (MCD). MCDs, like MCDDs, do not have directed circuits. They are finite graphs, obtained as a result of the instantiation of another finite graph.

6.4 Pathway Generation

Previously we have discussed how to add and build upon the modeling elements of CRL. Now we will demonstrate how to use CRL to perform various tasks. The remainder of this chapter will illustrate various features/properties of CRL, using case studies. We begin with an example from free radical chemistry. This domain has been chosen to test the efficacy of our approach given the myriad of pathways that free radicals can pursue. Let us consider the oxidation of butane.

CRL provides a utility for generating all possible reactions and identifying the resulting pathways, given a set of substrates and a reaction environment. The keyword arguments (i.e. arguments with a colon prefix) accepted by this method are given below:

(find-all-pathways :*substrates* :*operators* :*override-environment* :*initiator-p*)

The method tests for an override reaction environment allowing the user to investigate the influence of various environments without manually resetting the environment attribute of each substrate. [Recall that an abc (i.e. a substrate) knows its environment.] The user may also identify whether an initiator is present or believed to be present. This allows the user to investigate the effect of various reaction trajectories without knowing specifics concerning the initiator mechanism. The keyword :*operations* allows the user to specify the types of transformations to be used; the default value is **composite-operator**. The user may specify other operators as well. This focuses the elucidation of pathways. For example, by supplying $\mathbf{K_{free\text{-}radical}}$ to the keyword argument the user can investigate pathways of free radical origin.

Alternatively, the user may wish to investigate all theoretically feasible pathways subject to encoded preferences. This is accomplished by supplying :*operators* with $\mathbf{K^*}$, a modified composite operator. As described in chapter 5, $\mathbf{K^*}$ contains no encoded transformations within K_t but rather applies $\mathbf{K_{ai}}$ directly; and the K_f procedure of $\mathbf{K^*}$ reflects user-preferences exclusively. This allows the user to investigate pathways having prespecified features (i.e. $\Delta G < 10$ kcal/mol, stereocenters, etc.). Generation of these pathways is accomplished as follows:

(find-all-pathways :*substrates* (**butane oxygen**) :*operators* $\mathbf{K^*}$:*initiator-p* nil)

Similarly, if there are no preferences, theoretically feasible reactions can be generated by calling find-all-pathways with *:operators* **$K_{ab\text{-}initio}$** directly:

(find-all-pathways *:substrates* **(butane oxygen)** *:operators* **$K_{ab\text{-}initio}$**
:initiator-p nil)

This functionality allows the user to control the identification of potential pathways. Pathways connecting two states are identified using the method find-all-pathways-connecting. Its general form is:

(find-all-pathways-connecting : *substrates* *:desired-products*
:operators *:initiator-p*
:strategy)

where substrates may be a class allowing the user to investigate pathways leading from a desired product to a class or classes of substrates; alternatively, *:substrates* can be left unspecified. Once a control (e.g. synthesis) strategy has been specified, find-all-connecting-pathways chains on K_f and K_t, methods of **K**, using the semantic relation ***is-described-by*** and appropriate selector functions. The current implementation of CRL allows chaining only in the forward direction (i.e. synthetic direction) when **$K_{ab\text{-}initio}$** or **K^*** is supplied to *:operators*.

6.4.1 Pathway objects

Let us return now to the oxidation of butane. We will illustrate the generation of feasible pathways using **K^***. In this example, **K^*** was restricted to thermodynamically viable free radical pathways; products were restricted to their homologous series. The generation call was:

(find-all-pathways *:substrates* **(butane oxygen)** *:operators* **K^***
:initiator true)

Figure 6-8a presents several of the pathways identified when s-Bu is formed during the initiation process. The initiation reaction, **initiation-1**, for the oxidation of butane is presented in Figure 6-8b; therein, objects are denoted by #<object-name>. Recognize that each of these objects can in turn be expanded or disaggregated using the semantic relationships. Figure 6-8c illustrates one of the many pathways, **pathway-1**, generated during the oxidation of butane into s-BuOH and AcEt. The expansion of #<initiation-1>, an attribute value of composing-reactions, results in what is shown in Figure 6-8b. The user has access to this information at every step of the synthetic process using the semantic relationships provided by CRL.

By changing the **reaction-environment** new trajectories can be generated. These are denoted in Figure 6-9 using dashed lines and correspond to a higher energy environment. Figure 6-10 presents the trajectories of yet another **reaction environment**. The various pathways stemming from these environments are combined in Figure 6-11. The assumption set and conditions specifying each environment are managed by the context generation utility of CRL. [This utility will be discussed in detail in Section 6.4.2.]

By indexing and storing new reactions in their entirety (i.e. object storage), whether they be generated using **K** or discovered using $\mathbf{K}^*$ or $\mathbf{K_{ai}}$, we have given CRL the ability to learn. Moreover, since each reaction has a context, retrieval can be tailored with respect to interest (e.g. underlying assumptions) and sought transformation. [Reactions may also be searched by context.] This functionality allows CRL to take full advantage of past transformations.

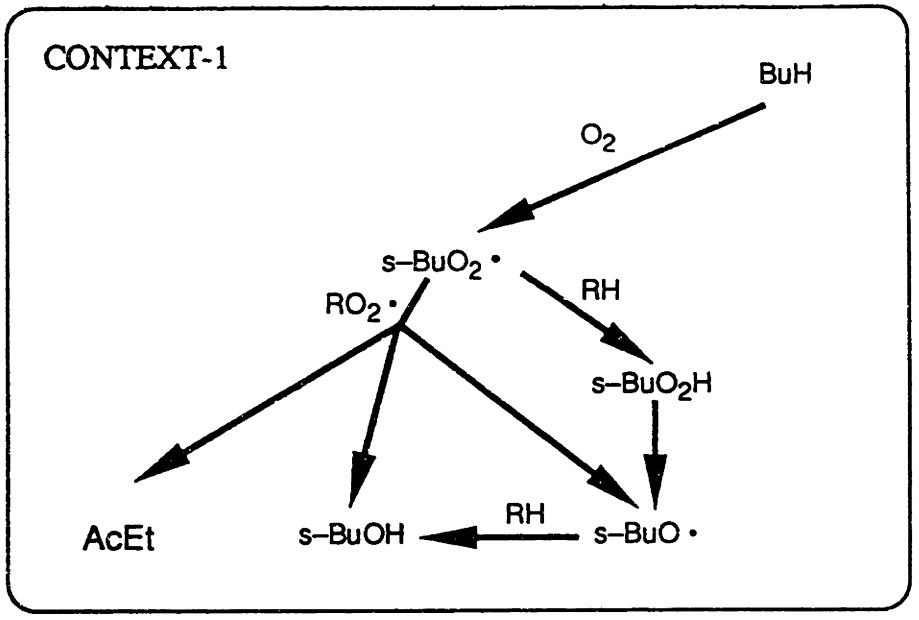


Figure 6-8a: Oxidation of butane

Reaction Object

identifier:	*unbound*
name:	initiation-1
reactants:	(#<$C_4H_{10}O_2$>)
products:	(#<C_4H_9O> #<HO>)
stoichiometry:	((#<$C_4H_{10}O_2$> . -1) (#<C_4H_9O> . +1) (#<HO> . +1))
reaction-environment:	#<reaction-environment-1>
enabling-conditions:	K_f
composing-transformations:	K_t
composing-reactions:	*unbound*
rate-expression:	#<rate-expression-1>
equilibrium-constant:	#<equilibrium-constant-1>
context:	#<context-1>

Figure 6-8b: Reaction object

Pathway Object

identifier	*unbound*
name	pathway-1
reactants	(# <C_4H_{10}> #<O_2>)
products	(#<$CH_3COCH_2CH_3$>)
stoichiometry	unbound
competing-pathways	(<pathway-2> #<pathway-3>)
composing-reactions	(#<initiation-1> #<initiation-2> #<abstraction-1> #<disproportionation-1> ...)
global-rate-expression	#<composite-rate-exp-1>
global-equilibrium-constant	*unbound*

Figure 6-8c: Pathway object

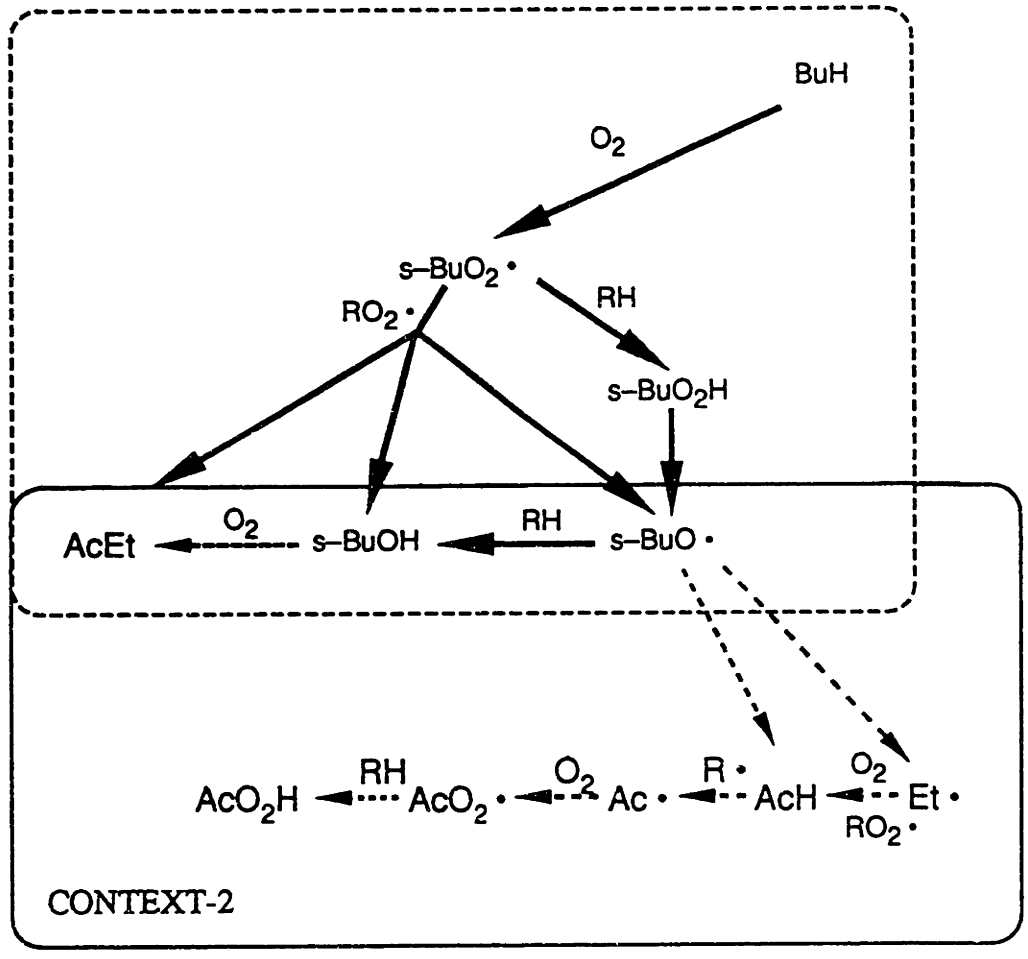


Figure 6-9: Oxidation of butane

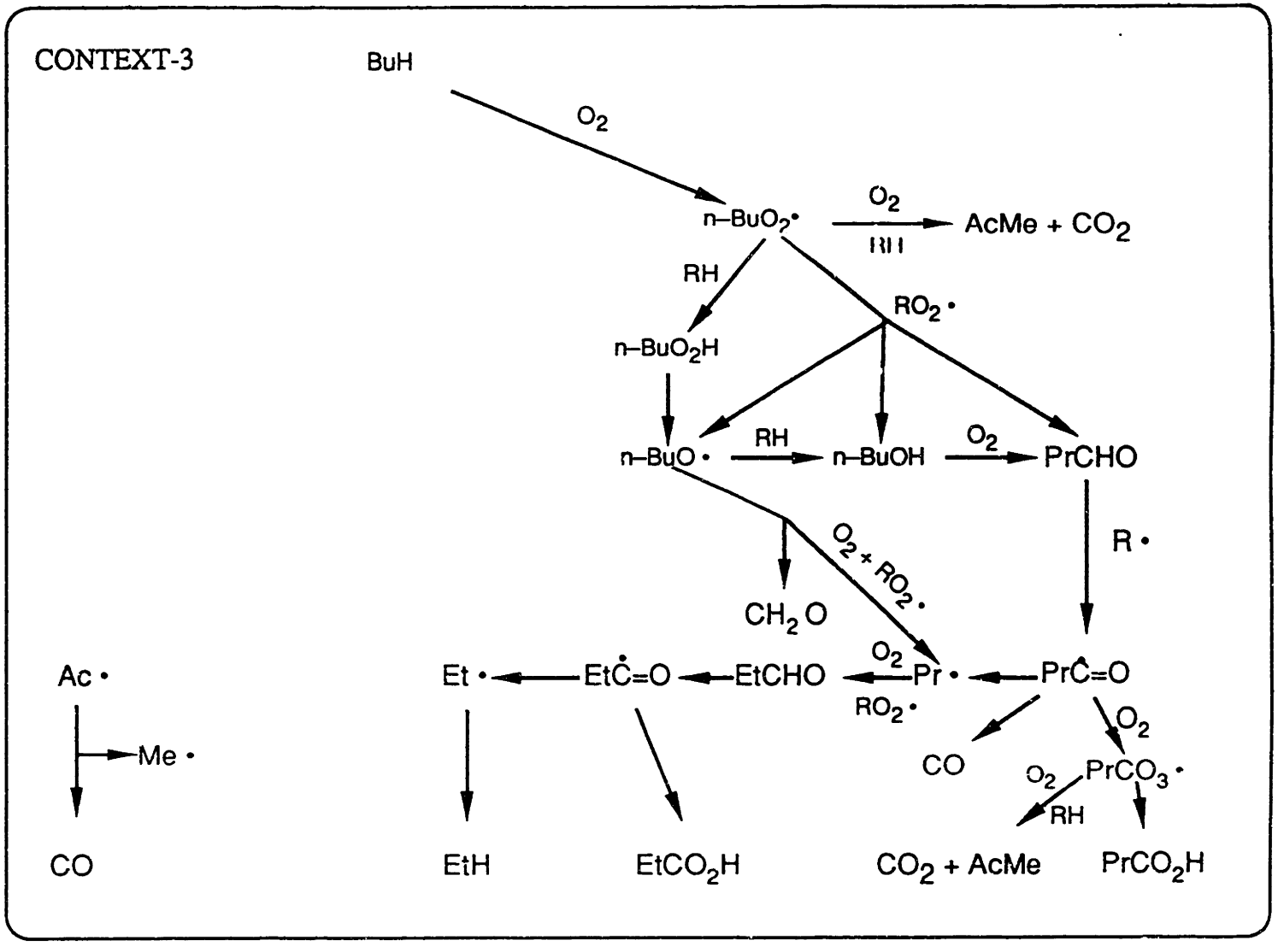


Figure 6-10: Oxidation of butane

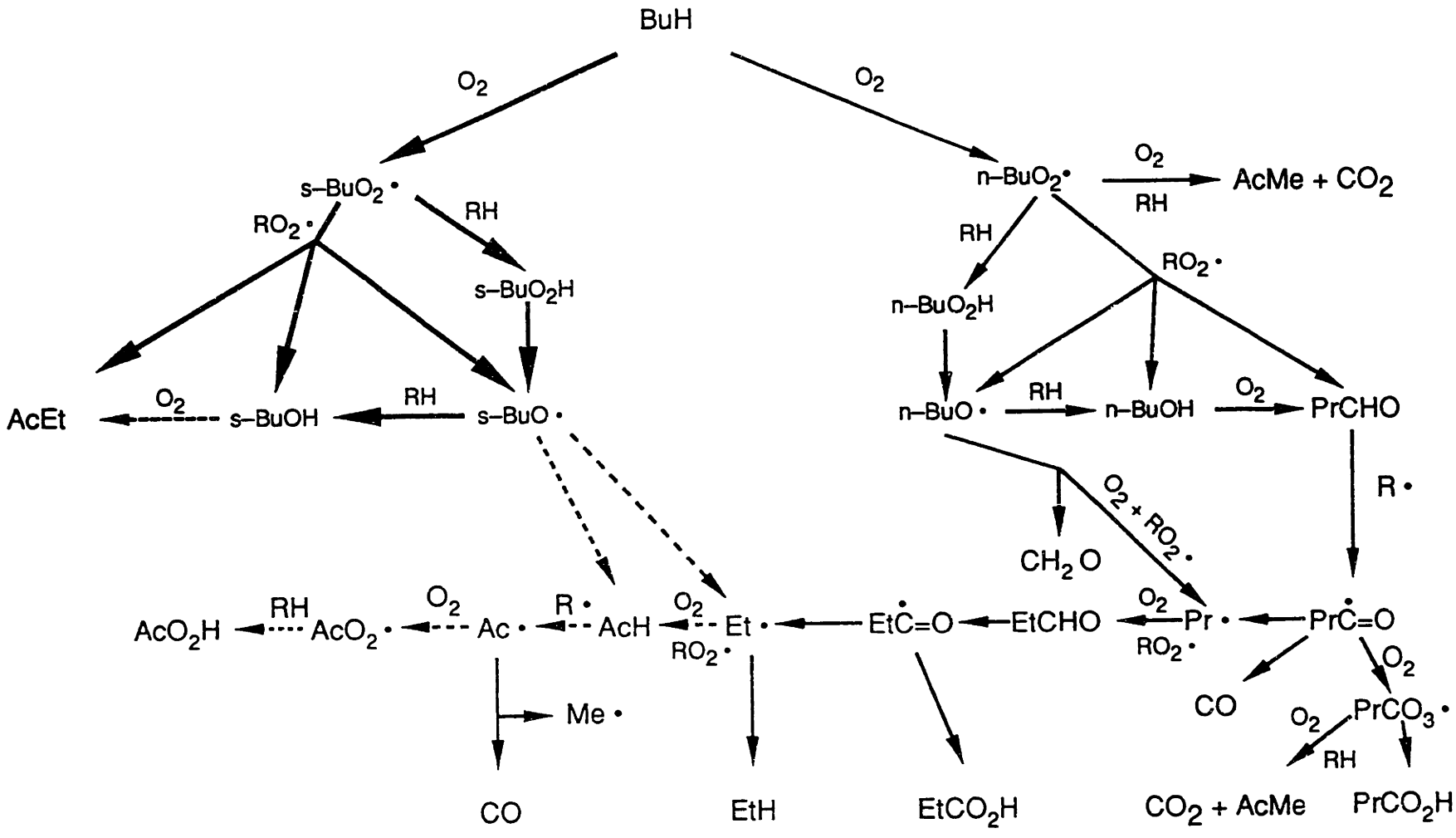


Figure 6-11: Oxidation of butane: linking contexts

6.4.2 Generation of contextual models

The description of CRL (see chapter 5) has shown that modeling involves the specification of structure and the description of the physicochemical phenomena taking place in it. These assumptions can be classified in the following categories:

- Assumptions on *structural components* that indicate component/subcomponent relations or legal combinations of subcomponents.
- Assumptions establishing the required relations among the various components (*functional constraints*), i.e. assumptions associated with constraints that represent balances, equilibrium conditions, chemical or transport rates and any other relations representing behavior or functionality.
- Assumptions creating the *interaction constraints*, that define the interactions among structural components.
- Assumptions on the *values of variables* that are describing attributes, i.e. assumptions that specify assignments of variable values.

6.4.3 Representation of modeling contexts

The above assumptions are expressed in terms of generalized constraints, which represent boolean, qualitative, semi-quantitative, or quantitative relationships among the modeling elements. The set of generalized constraints forms the context within which a model is valid, and are viewed as hard constraints taking on a boolean character, i.e. they are either satisfied or violated (never satisfied to a certain degree).

The generation of generalized constraints defining contextual models within CRL is driven by the set of established assumptions. Due to the multi-view characteristics of a model, the generalized constraints are quite different in character. For example, (a)

constraints generated from structural assumptions imply building a linking process, (b) functional constraints are semi-quantitative or quantitative relationships, etc.

When dealing with models at different levels of abstraction one defines distinct working contexts. Thus, the oxidation of butane case study has shown that each level is characterized with a set of assumptions, that in turn can be translated in terms of constraints.

But, in spite of the different assumptions and constraints, the contexts are not independent. For instance, the entities representing reactants in pathway-1 (i.e. reactants-1) are conceptually the same that the entities that represent reactants in pathway-2 (i.e. reactants-2). The same is valid for the variable objects that stand for conversion and selectivity in both contexts. In consequence, we need a mechanism to ensure context consistency during the whole synthetic process, that is. a way to inform entities located in different contexts of their conceptual equivalence. If the entities are complex, i.e. composite objects like pathways, reactions, relationships, etc., the mechanism should "propagate" the equivalency to the entities' descendants. Thus, if reactants-1 and reactants-2 are conceptually the same, the variables and relationships describing them should also know it.

The problem of dealing with models at multiple levels of abstraction can be viewed as a generalized constraint reasoning problem. In a simple constraint satisfaction problem, one tries to find consistent assignments of values to a given set of variables satisfying imposed constraints, where constraints and variables are all given in advance. In problems where models are represented at multiple levels, only a small set of assumptions, and thus constraints, will be valid in all different contexts, all others will be determined dynamically during the context building process.

6.4.4 Mechanism for generating contextual models

Let us assume that a model X developed within CONTEXT-1, is to be modified. CONTEXT-2 is created as a "child" of CONTEXT-1, that will be referred as to the parent-context. In principle, CONTEXT-2 "inherits" all the assumptions that are valid in CONTEXT-1. Then, in order to account for the modifications of model X it is possible to change the inherited assumptions with additions and deletions. The changes are covered by the following rule:

> "All assumptions in CONTEXT-1 are "inherited" by CONTEXT-2 unless overridden in CONTEXT-2".

Generalizing this rule we can say that:

> "The assumptions true in a given context are all the assumptions that are true in the parent-context, minus the assumptions that have been specifically deleted in it, plus any assumption that has been specifically added in it".

Following with this hypothetical case, let us assume that a different representation of X is required. Now, two options are given to us: (i) If the new representation is seen as an alternative to the one already introduced in CONTEXT-2, a new context, called CONTEXT-3, should be created as a child of CONTEXT-1. (ii) If the modification is intended to change model X built in CONTEXT-2, CONTEXT-3 should be created as a child of CONTEXT-2. The first choice is analogous to the situation found in Butane Oxidation (Figure 6-9). The assumptions corresponding to the free radical oxidation of butane are encapsulated in CONTEXT-1. Given that two distinct global pathways have to be analyzed, CONTEXT-1 gives rise to CONTEXT-2 and CONTEXT-3 and encapsulate the assumptions that generate the representations of Figures 6-9 and 6-10, respectively. CONTEXT-2 and CONTEXT-3 are siblings: both have information in common with CONTEXT-1, but there is no direct relation between them. The second

situation corresponds to many multistep syntheses. During the synthesis of a complex molecule the chemist makes assumptions regarding how the molecule should be constructed; and these assumptions give rise to new assumptions. For example, assumptions regarding the specification of a carbon skeleton with correct key group orientation may be kept in context #1. This context may give rise to context #2 by placing restrictions (i.e. assumptions) on the latent functionality of the skeleton; and these restrictions may give rise to context #3: assumptions regarding the need for certain protective groups.

Generalizing, we can state that contexts are always arranged in hierarchical manner. The relations among them are established in a directed graph generated by the context relation (CR) on the set of contexts, C. The relation CR is read as "*gives-rise-to*", and the digraph is called MODEL ABSTRACTION TREE (MAT). It is a tree because each node representing a context has only one predecessor. A MAT is defined by

$$MAT = (C, U_R)$$

where

$$C = \{CONTEXTS\}$$

$$U_R = \{(C_i, C_j) \in C \times C / C_i \; CR \; C_j\}$$

This tree of contexts is similar to the "class precedence list" used in hierarchical inheritance mechanisms. An example of such tree is presented in Figure 6-12, where CONTEXT 1 gives rise to CONTEXT 2 and 3, CONTEXT 2 gives rise to CONTEXT 4 and 5, and CONTEXT 3 to CONTEXT 6 respectively.

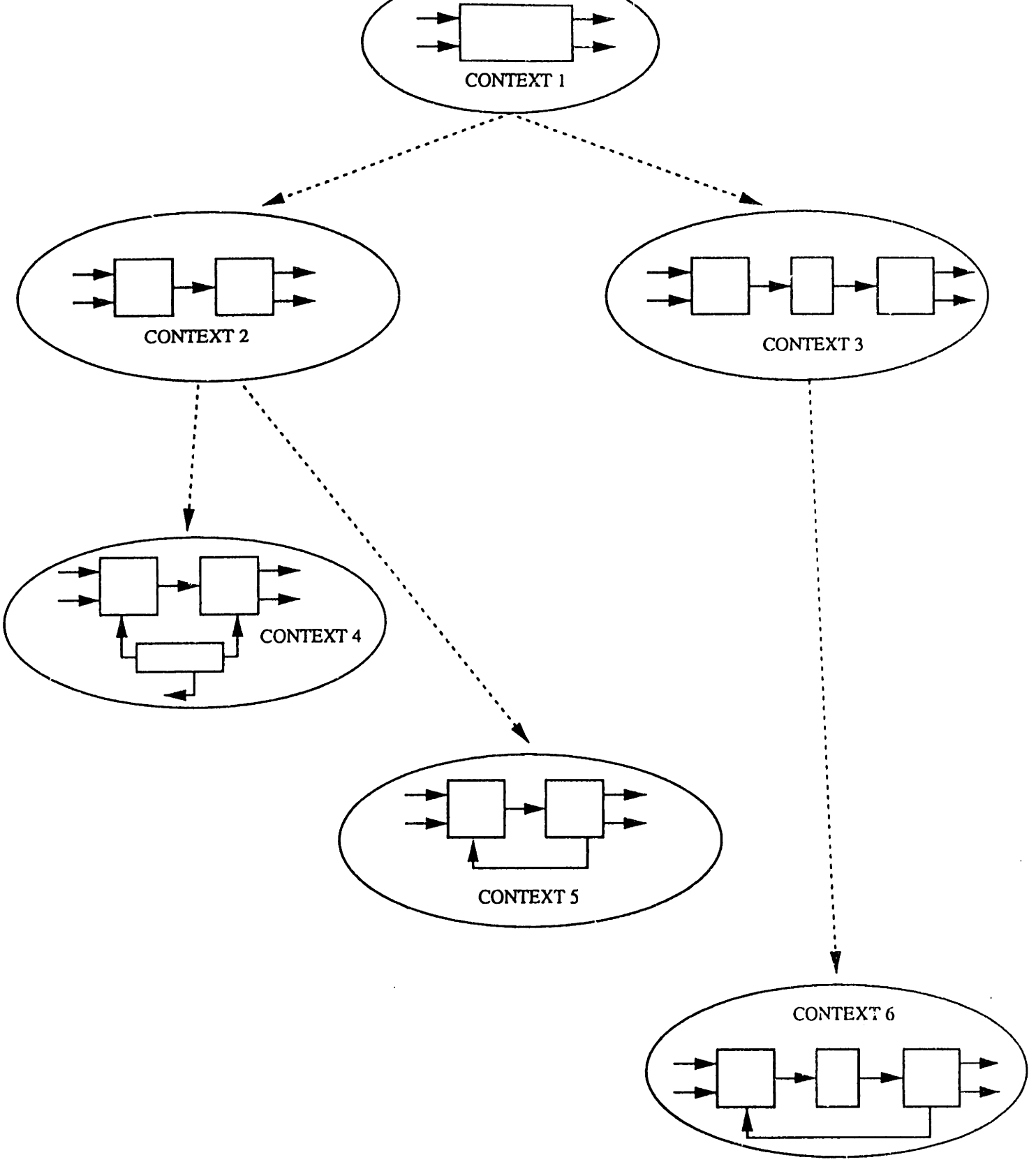


Figure 6-12: Representation of a model abstraction tree

6.4.5 Communication of contextual models

Contexts can only be related as:

(a) siblings (e.g. CONTEXT 2 and CONTEXT 3 in Figure 6-11)

(b) "parent"-"child" (e.g. CONTEXT 1 and CONTEXT 2 in Figure 6-12).

In case (a), the contexts do not need to communicate with each other, but in case (b) information must pass from the parent-context to the child-context. The operations associated with the modification of the parent-context to produce the child-context, presently available in CRL are:

MODIFY: an operation applied to an individual assumption in order to alter its value.

DELETE: an operation applied to a single assumption in order to alter its value.

ABSTRACT: an operation that is applicable to the set of assumptions representing an object model. The representation of this object is abstracted to obtain a less detailed description of its structure.

DISAGGREGATE: is the opposite to ABSTRACT. It is applied to the assumption set, representing a model, but its purpose is to introduce more refinement.

While MODIFY and DELETE are simple operations, ABSTRACT and DISAGGREGATE are quite complex. Whichever of the last two operations is performed during the modification of a new context, it should allow the user to bind together components that are conceptually related in the two contexts. Thus, the ABSTRACT and DISAGGREGATE operations should provide the framework for the communication across contexts. The symmetric operations of abstraction and

disaggregation can be described by an algorithm that considers them in a very general sense, and can be summarized as follows:

```
ABSTRACTION (NEW-CONTEXT NEW-UNIT OLD-UNITS)

    ESTABLISH-STRUCTURAL-COMPATIBILITY (NEW-CONTEXT NEW-UNIT
    OLD-UNITS)

    ESTABLISH-PHENOMENA-COMPATIBILITY (NEW-CONTEXT NEW-UNIT OLD-
    UNITS)

END


DISAGGREGATION (NEW-CONTEXT NEW-UNITS OLD-UNIT)

    ESTABLISH-STRUCTURAL-COMPATIBILITY (NEW-CONTEXT NEW-UNITS
    OLD-UNIT)

    ESTABLISH-PHENOMENA-COMPATIBILITY (NEW-CONTEXT NEW-UNITS
    OLD-UNIT)

END
```

Since contexts have only one parent there is no need to specify the name of the context where the old-units are located. With the name of a new context, its parent context will be uniquely specified. The details of the above algorithms will be discussed in the following sections, where we will make extensive use of the semantic links ***is-disaggregated-in*** and ***is-abstracted-by*** which were introduced in chapter 5.

6.5 Concurrent Models at Multiple Levels of Abstraction

In this section we will discuss the mechanism that CRL uses to generate consistent, concurrent models at different levels of modeling abstraction. More specifically, we will examine the procedures used to implement the ABSTRACTION, DISAGGREGATION operations introduced in the previous section.

6.5.1 Establish-structural-compatibility

Structural compatibility analysis operates on pathways (**Ps**) that are located in two different contexts bound together by a CR relation. During the disaggregation operation, a **P** that exists in the parent context is substituted by a set of **Ps**, in the new context. The structural compatibility operation will establish the following semantic link:

$\mathbf{P_{old}}$ ***is-disaggregated-in*** (SET.OF ($\mathbf{P_1}$,..$\mathbf{P_n}$,)) in NEW-CONTEXT

Since a parent-context may have several children, it is always necessary to specify the name of the child-context when the disaggregation occurs. For example, during the oxidation of butane, the context **butane-oxidation-reaction-environment** gives rise to the contexts **butane-oxidation-termination-1** and **butane-oxidation-termination-2**. So, the relation:

butane-pathways ***is-disaggregated-in***
(SET.OF (**initiation, propagation, termination**))
in **butane-oxidation-termination-1**

provides an unambiguous link and avoids any confusion with models located in the other contexts (**initiation, propagation**, ..., etc.)

The abstraction procedure is just the inverse operation:

(SET.OF ($\mathbf{P_1}$,...$\mathbf{P_n}$,)) ***is-abstracted-by*** $\mathbf{P_{new}}$ in NEW-CONTEXT

Now, a set of **Ps** are abstracted in a single pathway.

Figure 6-13a shows an example of conflicting structural assumptions between CONTEXT-i and CONTEXT-j. If CONTEXT-j is created as a child of CONTEXT-i, the structural-compatibility operation will generate the link

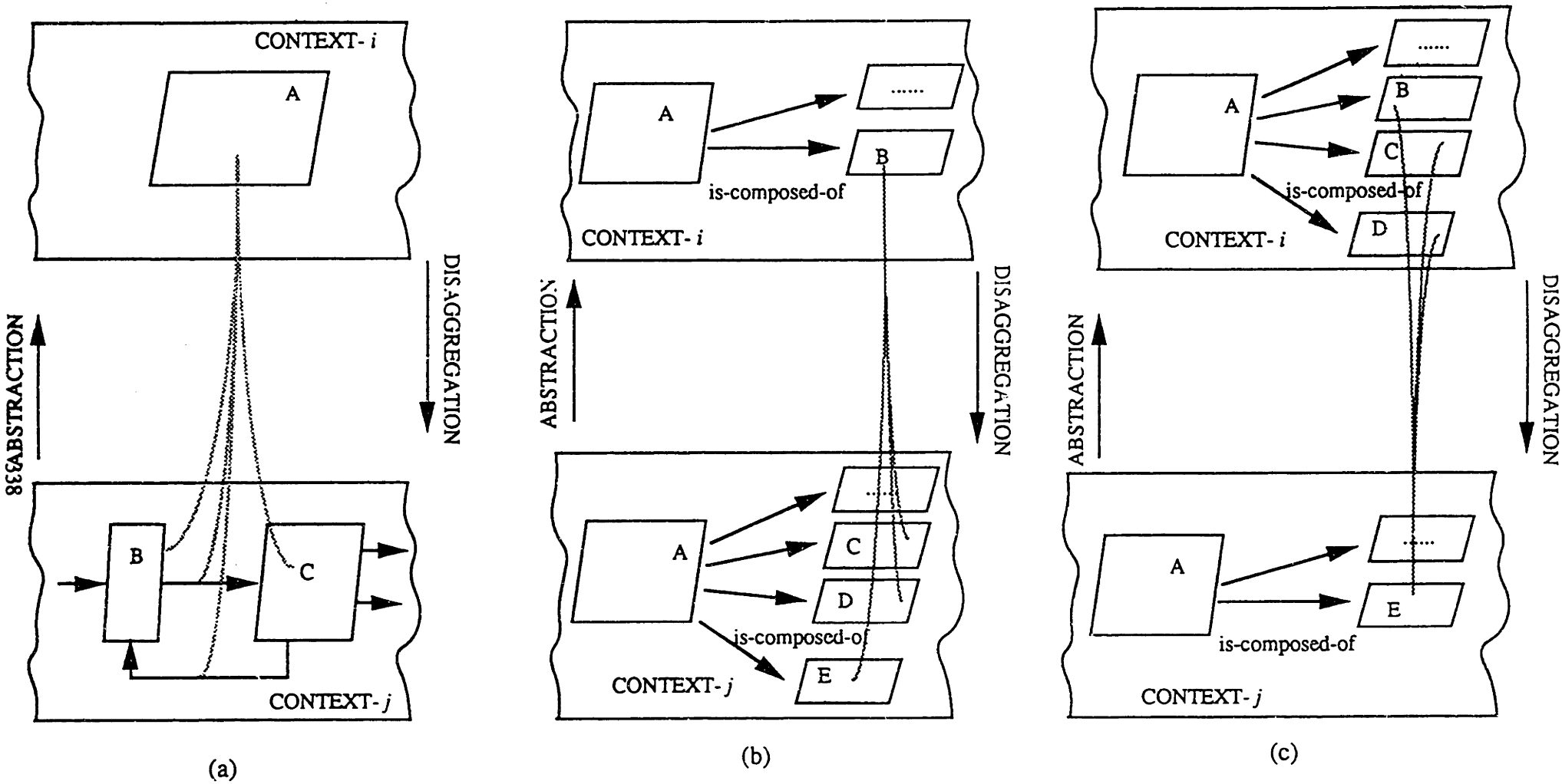


Figure 6-13: (a) Confliction structural assumptions between CONTEXT-i and CONTEXT-j. Meaning of the ABSTRACTION and DISAGGREGATION operations. (b) Reestablishing structural consistency after the disaggregation of a pathway that is-part-of another pathway. (c) Reestablishing structural consistency after the abstraction of a set of pathways that are comprising another pathway.

A ***is-disaggregated-in*** (SET.OF (B, C)) in CONTEXT-j

On the other hand, if CONTEXT-i is a child of CONTEXT-j, the structural compatibility operation would generate the following link

(SET.OF (B, C)) ***is-abstracted-by*** A in CONTEXT-i

However, there are more complex situations that need further analysis. Specifically, the case of pathways which are being disaggregated or abstracted and that themselves are not simple pathways from a structural point of view, but are also "part-of" other pathways in the parent context. What happens with these semantic links in the new context? A typical situation is depicted in Figure 6-13b, where pathway B, that ***is-part-of*** pathway A in CONTEXT-i, ***is-disaggregated-in*** pathways C, D and E in CONTEXT-j. The structural-compatibility procedure, apart from establishing ***is-disaggregated-in*** links across the contexts, should also establish ***is-part-of*** links between pathways C, D and E, and pathway A.

More formally, the structural-compatibility procedure should perform the following checking operation every time a pathway $\mathbf{P_{old}}$ is disaggregated in a new context

IF

$\mathbf{P_{old}}$ ***is-part-of*** ?PATHWAY-X in OLD-CONTEXT

THEN

?PATHWAY-Y ***is-part-of*** ?PATHWAY-X in NEW-CONTEXT

where ?PATHWAY-Y is one of the pathways in which $\mathbf{P_{old}}$ is disaggregated; that is, the following link exists between them

$\mathbf{P_{old}}$ ***is-disaggregated-in*** ?PATHWAY-Y in NEW-CONTEXT

In order to avoid the generation of redundant links that the checking operation can produce due to the transitivity axiom, ?PATHWAY-X will be restricted to be the predecessor of $\mathbf{P_{old}}$ that is related to $\mathbf{P_{old}}$ by means of a ***is-composed-of*** link. This condition is met by the only member of the following set

$$\{x/x \in A \wedge \{ \{y/x \textit{ is-composed-of } y\} \cap A\} = \phi\}$$

where

$$A = \{x/x \textit{ is-composed-of } \mathbf{P_{old}}\}$$

Let us now consider the inverse situation, depicted in Figure 6-13c, where pathways B, C and D, that are part of pathway A in CONTEXT-i are abstracted in pathway E in CONTEXT-j. In this case, the structural-compatibility procedure will establish ***is-abstracted-by*** links across the contexts and a ***is-composed-of*** link between pathways A and E in CONTEXT-j. In order to consider situations like the one presented above, the following checking operation is also carried out every time a set of pathways $\{\mathbf{P_i}, i=1,.., n\}$ ***is-abstracted-by*** the pathway $\mathbf{P_{new}}$ in a new context.

IF

?PATHWAY-X ***is-composed-of*** $\{\mathbf{P_i}, i=1,.., n\}$ in OLD-CONTEXT

THEN

?PATHWAY-X ***is-composed-of*** $\mathbf{P_{new}}$ in NEW-CONTEXT

where $\{\mathbf{P_i}, i=1,.., n\}$ ***is-abstracted-by*** $\mathbf{P_{new}}$ in NEW-CONTEXT.

For the same reasons expressed above, that is to avoid the creation of redundant links, ?PATHWAY-X will be restricted to be the pathway which is a predecessor of the members of the set $\{\mathbf{P_i}, i=1,.., n\}$, and which is also related to them by a ***is-composed-of*** link. Like before, ?PATHWAY-X is the only member of the following set

$$\{x/x \in B \wedge \{ \{y/x \textit{ is-composed-of } y\} \cap B\} = \phi\}$$

where

$$B = \{x/x \textit{ is-composed-of } \{P_i, i=1,.., n\}_{old}\}$$

6.5.2 Propagation of variable values across contexts

Up to this point the operations of abstraction and disaggregation have only implied the creation of new links between pathways representing similar concepts in different contexts. However, these links should be propagated to the variables that are describing the corresponding objects in order to allow the passing of information. In fact, the ultimate goal of the consistency operations is to make possible the propagation of variable values across contexts, by providing the formal framework in which the propagation should be performed.

Linking operations. Thus, once two or more pathways are related by an abstraction/disaggregation link, we want an automatic procedure to relate the variables describing them. As with the objects seen before, the variables can be linked by one-to-one or one-to-many relations. The procedure that performs the linking operation can be summarized as follows:

IF

(?OBJECT-X *semantic-link* ?OBJECT-Y) IN NEW-CONTEXT

(?VAR-X *is-describing* ?OBJECT-X) IN NEW-CONTEXT

(?VAR-Y *is-describing* ?OBJECT-Y) IN OLD-CONTEXT

(?VAR-X *is-a-member-of* ?VAR-CLASS)

(?VAR-Y *is-a-member-of* ?VAR-CLASS)

(UNAMBIGUOUS ?VAR-X NEW-CONTEXT)

(UNAMBIGUOUS ?VAR-Y OLD-CONTEXT)

THEN

(?VAR-X **semantic-link** ?VAR-Y) IN NEW-CONTEXT

END

Ambiguity checking operations. Given two or more objects related by an ***is-disaggregated-by*** or an ***is-abstracted-by*** link, the procedure presented above is applied to the objects themselves, as well as to their inferiors, that is to the objects having ***is-part-of*** or ***is-attached-to*** links with them. An analysis of this procedure shows that in order to link variable objects, they have to be variables of the same type. Furthermore, they have to be unambiguous, that is, if we have two variables of the same type describing the same object (e.g. two reaction rates describing a given reaction) it will not be possible to establish automatically links with the conceptually equivalent variables located in the other context. In these cases the user will resolve the ambiguity by establishing the mapping. The ambiguity checking procedure can be summarized as follows:

```
UNAMBIGUOUS? (VAR CONTEXT?
    IF
        (VAR          is-describing  ?OBJECT-X) IN CONTEXT
        (?VAR-Y       is-describing  ?OBJECT-X) IN CONTEXT
        (VAR          is-a-member-of      ?VAR-CLASS)
        (?VAR-Y       is-a-member-of      ?VAR-CLASS)
        (?VAR-CLASS   is-a  GENERIC-VARIABLE)
    THEN
        (RETURN FALSE)
    ELSE
        (RETURN TRUE)
    END IF
END UNAMBIGUOUS
```

Once the link between variable objects has been established the route to pass information is defined. However, variable values should not be propagated automatically, this is a process that should be driven by the user. [The user can be a person or a program sitting on top of the modeling system.] The user defines contexts, by encapsulating conflicting assumptions, and decides when is proper to reestablish consistency in terms of assumptions on values of variables.

Transfer of values in one-to-one links. When two variables are related by a one-to-one relation, the user can directly transfer information between them, by specifying the location to which the value will be passed. For example, let us consider the case depicted in Figure 6-14 where the pathway is related to the reaction-set by one-to-one links. Consequently, when the procedure presented above is applied to them, the variables describing conceptually equivalent entities will get linked by one-to-one relations like:

Substrates-P-1 *is-disaggregated-in* Reactants-A-R-1, Reactants-B-R-1

Concentration-Substrates-P-1 *is-disaggregated-in* Concentration-Reactants

- - - - - - - - - - - - - - -

- - - - - - - - - - - - - - -

PRODUCT-P-1 *is-disaggregated-in* Product-R-2

BYPRODUCT-P-1 *is-disaggregated-in* By-Product-R-2

So, if the user decides to transfer specific variable values from the Pathway-structure-context to the Reaction-set-context, statements like:

OBTAIN-VALUE-OF PRODUCT-R-2 **IN** Reaction-Set-Context

FROM Pathway-Structure-Context

OBTAIN-VALUE-OF CONCENTRATION-BY-PRODUCT-R-2 **IN**

Reaction-Set-context **FROM** Pathway-Structure-Context

will need to be introduced.

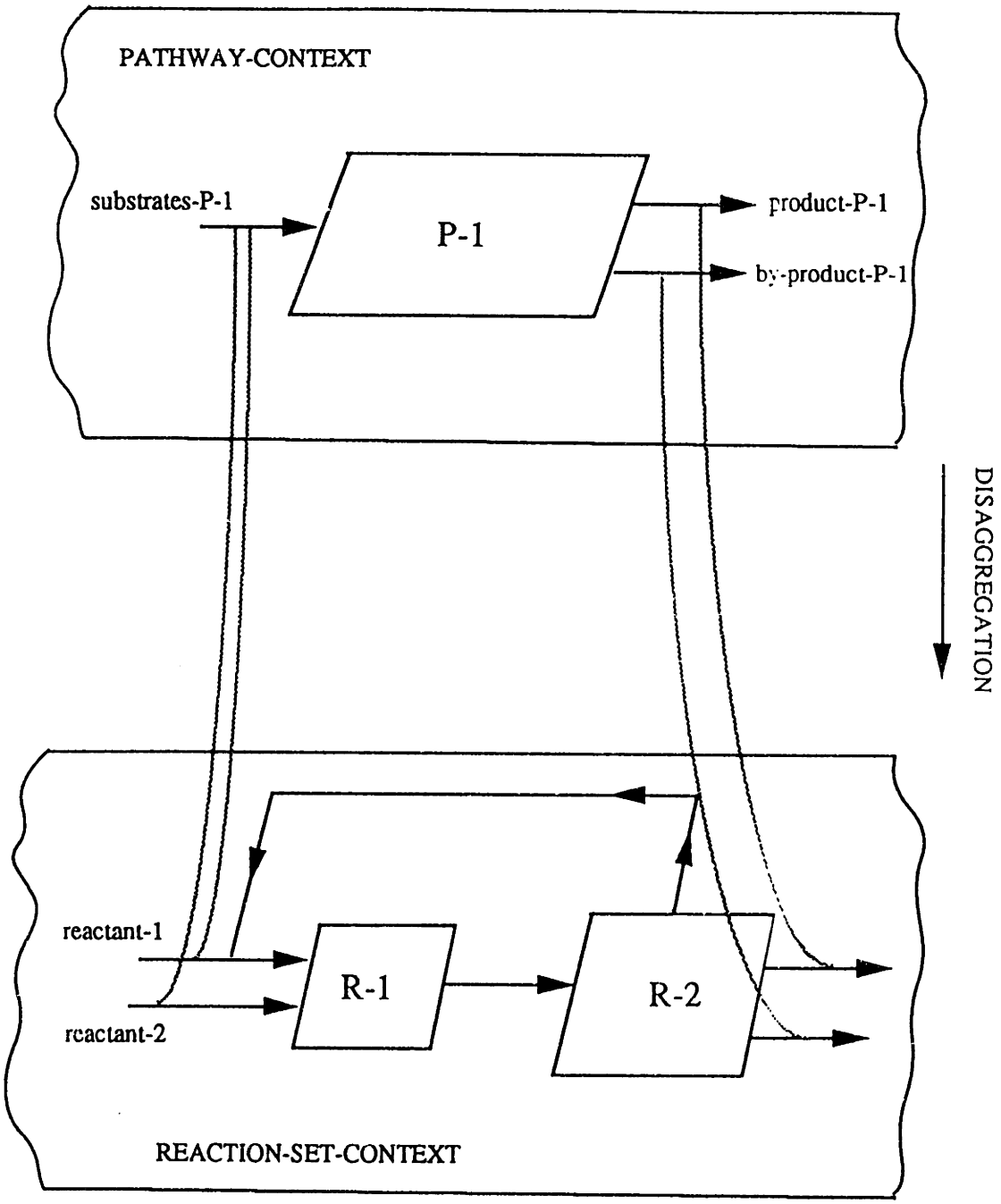


Figure 6-14: Disaggregation of PATHWAY-CONTEXT into REACTION-SET-CONTEXT

Transfer of values in one-to-a-set links. However, when a variable located in one context is related to several variables in another context (i.e. one-to-a-set link) the transfer of information is not straight forward. In this case, the user needs to define a mapping function like:

$$\text{X-VALUE} = \text{MF}_{\text{CONTEXT-j}}(\text{Y-VALUES})$$

that relates the value of the variable object X with the values of the set of variable objects Y, where X and the vector Y are related through

$$\text{X } \textit{is-disaggregated-in } (\text{SET.OF } (Y_i, i = 1, n))$$

This implies that the application of MF on the vector Y will give the value of X in the context that is related to CONTEXT-j where X exists. In order to illustrate these ideas let us consider the simple case presented in Figure 6-15. In this example we have that

PRODUCTS-A *is-disaggregated-in* {PRODUCTS-C, PRODUCTS-D} in CONTEXT-j

This relation implies (according to the linking operations presented earlier) that the variables describing the ports are linked by relations like:

CONCENTRATION-PRODUCTS-1 *is-disaggregated-in* {CONCENTRATION-PRODUCTS-C CONCENTRATION-PRODUCTS-D in CONTEXT-j

If the user wants to move values between CONTEXT-i and CONTEXT-j, appropriate mapping function must be defined. In this example:

$$\text{COMPOSITION-PRODUCTS-A-VALUE} = \text{MF}_{\text{CONTEXT-j}}(\text{COMPOSITION-PRODUCTS-C-VALUE COMPOSITION-PRODUCTS-D-VALUE})$$

$$\text{MF}_{\text{CONTEXT-j}} = \text{COMPOSITION-PRODUCTS-C-VALUE} + \text{COMPOSITION-PRODUCTS-D-VALUE}$$

Once the semantic links and the MFs have been established, we will use again a statement of the following type

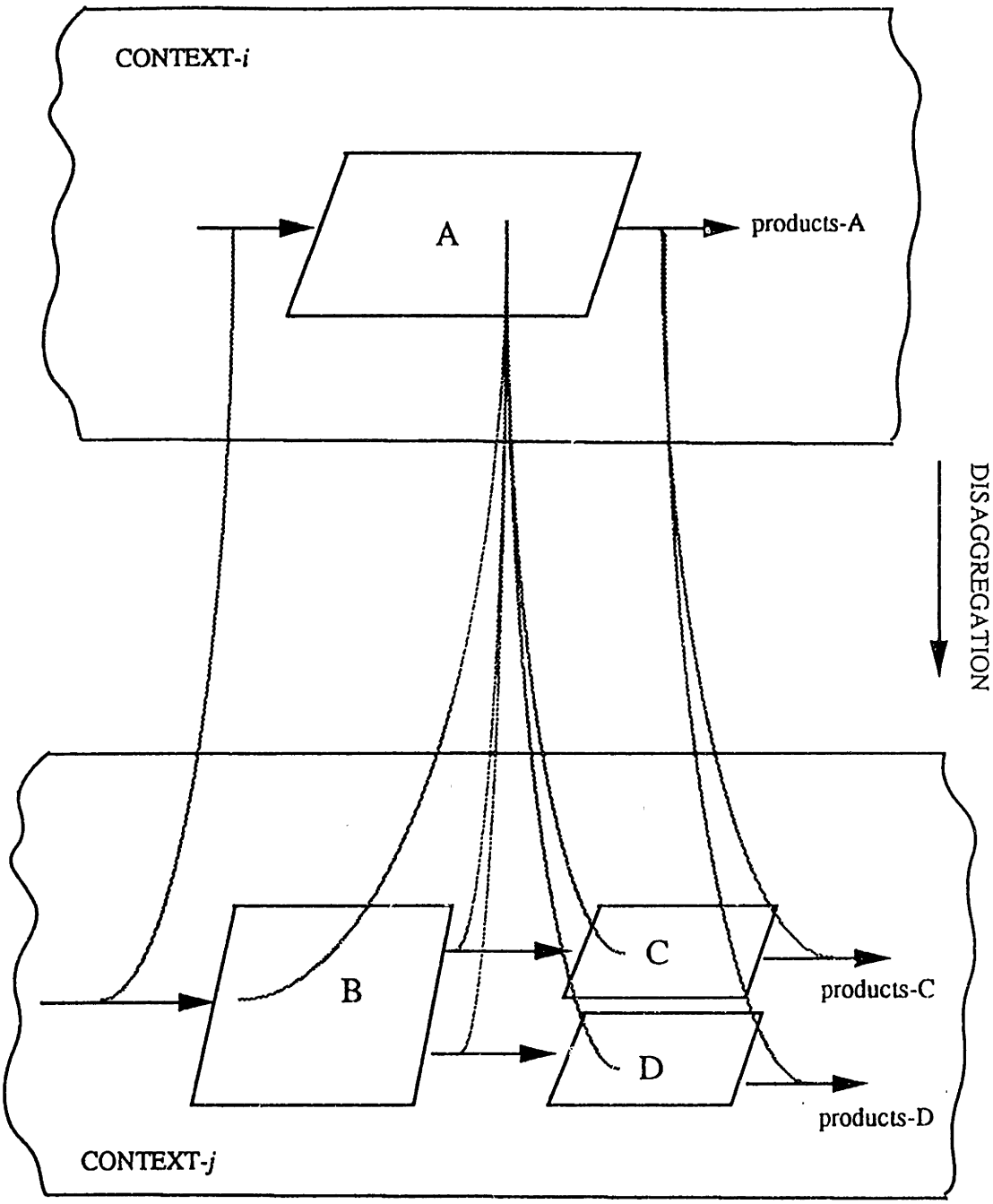


Figure 6-15: Pathway A, located in CONTEXT-i, is disaggregated in CONTEXT-j. Links are established between the conceptually equivalent entities.

(OBTAIN-VALUE-OF VARIABLE IN CONTEXT-i **FROM** CONTEXT-j)

to transfer the information. For instance, to deduce the value of COMPOSITION-PRODUCT-A in CONTEXT-i using information stored in CONTEXT-j we have to write

(OBTAIN-VALUE-OF COMPOSITION-PRODUCTS-A **IN** CONTEXT-i **FROM** CONTEXT-j)

When this simple statement is parsed, the OBTAIN-VALUE procedure will be fired. This procedure takes three arguments: The name of the variable that we want to give a value, the name of the context in which it is located, and the name of the context from which we want to get the information. Up to now, in the propagation of variable values across contexts, we have assumed that the two contexts that are involved have a parent-child relation. However, this condition is too restrictive; variable values can be propagated between two contexts provided that there is a path between them in the MAT (model aostraction tree). Considering the example of Figure 6-12, we can transfer information between contexts #1 and #5, and contexts #1 and #6. However, the non-existence of a path between contexts #5 and #6, eliminates the possibility of communication between them. With these ideas in mind we can give a broader interpretation to the OBTAIN-VALUE statement. In fact, the OBTAIN-VALUE is a very general procedure, that will check all the necessary prerequisites to transfer information between variables, and is applicable to variables related by one-to-one or one-to-a-set links, that are located in two contexts such that there is a path between them. The details of the OBTAIN-VALUE algorithm are given in Appendix B.

An analysis of the concepts presented above indicates that the one-to-one link between variable objects is a special case of the one-to-a-set type of link. When the links is one-to-one, the mapping function does not need to be specified by the user because the MF is trivial, it only equates variable values. This can also be the case of certain one-to-a-set

links involving extensive variables. For example, the treatment of a variable like flowrate is obvious, and the knowledge about its associated MF can be easily incorporated into the system, avoiding the need for a definition provided by the user. It should be emphasized again that when we refer to "the user", the word user always means a person or a program. Finally, we want to make clear the fact that MF can involve any type of relation. The condition is that it should be consistent with the type of content (real number, integer, boolean, qualitative, etc.) of the variables that it comprises. That is, quantitative relations will be used to propagate quantitative values, qualitative relations to propagate qualitative values, etc.

6.6 Illustrations

The focus of our endeavors has been on the development of a modeling language to ameliorate computer aided chemical reasoning. We believe the structure and foundations of this language provide the expressiveness necessary to support diverse applications in a wide range of fields (see Section 6.1.1).

Due to myriad of pathways which free radicals can pursue, this domain has been chosen to test the efficacy of our approach. We have elucidated the free radical pathways of various hydrocarbons, peroxides, and polymers for evaluation purposes. Our concentration has been on autooxidation and halogenation reactions due to their potentially hazardous nature. In this spirit, we offer examples illustrative of the program's functionality. Herein CRL methods are represented in SMALL CAPS.

6.6.1 Case Study 1: ethane pyrolysis

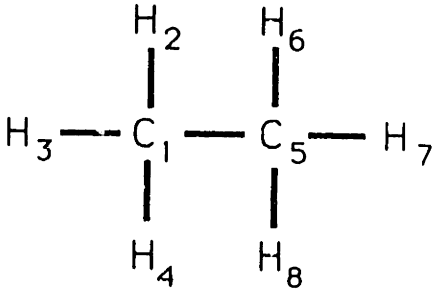
To demonstrate the utility of CRL and the interaction between various modeling elements, semantic relationships, and supporting methods, let us first consider the pyrolysis of ethane forming principally ethylene and hydrogen. Although this example is

relatively simple, it highlights the funtionality of CRL and underscores various issues which require resolution for computer implementation to be successful.

We begin by initializing the **abc** to represent **ethane**. Since ethane does not contain stereo centers, initialization to form the object graph representation can be accomplished directly from the bond electron matrix representation (be-matrix) shown in Figure 6-16. This representation is an extension of Spialter's atom connectivity matrices (ACM) first introduced in 1963 for documentation purposes [16] and although ACMs are sufficient for these purposes, they do not adequately represent interconvertable chemical systems. Systems of this type require double bond keeping capability to account for both for both bonds and electrons. This can be achieved by combining the ACM with valence electrons corresponding to each atom in the matrix along the diagonal resulting in a compact representation adequate for closed shell chemistry: chemistry involving integer bond numbers and uncharged atoms.

We have enhanced this representation by converting it into an object graph, wherein the nodes are **atoms** and the edges are **bonds**, so that additional knowledge can be associated with these elements allowing fuller description. Transformation is accomplished by parsing the bond electron matrix using MAKE-GRAPH-FROM-SYMMETRIC-CONNECTIVITY-MATRIX followed by applying MAKE-ATOM-BOND-CONFIGURATION to the resulting graph forming the **abc** shown in Figure 6-17; objects are denoted with a # prefix. Methods associated with the underlying **abc** modeling element provide the vehicle for establishing attribute values describing the hydrocarbon **ethane**.

Each of these attributes whose values is an object can in turn be expanded using the semantic relationships provided by CRL. For example, expansion of **db-atom**, makes accessible properties that describe the atom which are independent of its environment (see Figure 6-18). These may include: electronegativity, valence electrons, atom-



	1	2	3	4	5	6	7	8
1	0	1	1	1	1	0	0	0
2	1	0	0	0	0	0	0	0
3	1	0	0	0	0	0	0	0
4	1	0	0	0	0	0	0	0
5	1	0	0	0	0	1	1	1
6	0	0	0	0	1	0	0	0
7	0	0	0	0	1	0	0	0
8	0	0	0	0	1	0	0	0

Figure 6-16: Ethane bond electron matrix

```
identifier:                    "C2H6-T0779"
name:                          "ethane"
atoms:                         (#<ATOM C-T0764> #<ATOM H-T0765>
                                #<ATOM H-T0766> #<ATOM H-T0767>
                                #<ATOM C-T0768> #<ATOM H-T0769>
                                #<ATOM H-T0770> #<ATOM H-T0771>)
bonds:                         (#<BOND b-T0772>  #<BOND b-T0773>
                                #<BOND b-T0774>  #<BOND b-T0775>
                                #<BOND b-T0776>  #<BOND b-T0777>
                                #<BOND b-T0778>)
empirical-formula:             ((#<DB-ATOM 414000575> . 6) (#<DB-ATOM 414001073> . 2))
empirical-formula-string:      "C2H6"
molecular-weight:              30.07
charge:                        0.0
terminal-skeleton-atoms:       (#<ATOM C-T0764>  #<ATOM C-T0768>)
equivalent-atoms:              ((#<ATOM C-T0768>  #<ATOM C-T0764>)
                                (#<ATOM H-T0771>  #<ATOM H-T0765>
                                #<ATOM H-T0766>  #<ATOM H-T0767>
                                #<ATOM H-T0769>  #<ATOM H-T0770>))
equivalent-bonds:              ((#<BOND b-T0775>)
                                (#<BOND b-T0778>    #<BOND b-T0772>
                                #<BOND b-T0773>    #<BOND b-T0774>
                                #<BOND b-T0776>    #<BOND b-T0777>))
weakest-bond:                  (#<BOND b-T0775>)
weakest-bond-strength:         82.6
weakest-bond-strength-ratio:   1.0
ordered-eq-bonds:              ((#<BOND b-T0775>)
                                (#<BOND b-T0778>    #<BOND b-T0772>
                                #<BOND b-T0773>    #<BOND b-T0774>
                                #<BOND b-T0776>    #<BOND b-T0777>))
groups:                        (#<GROUP methyl-1>  #<GROUP methyl-2>
                                #<GROUP terminal-sp3-methylene-1>
                                #<GROUP terminal-sp3-methylene-2>)
group-bonds:                   (#<BOND-b-T0775>    #<BOND b-T0772>
                                #<BOND b-T0776>)
meta-groups:                   (#<GROUP ethyl-1>    #<GROUP ethyl-2>)
methyl-carbon:                 NIL
primary-carbons:               (#<ATOM C-T0764>  #<ATOM C-T0768>)
secondary-carbons:             NIL
tertiary-carbons:              NIL
terminal-carbons:              (#<ATOM C-T0764>  #<ATOM C-T0768>)
backbones:                     (#<ATOM C-T0764>  #<ATOM C-T0768>)
backbone-length:               2
progenitor:                    NIL
environment:                   #<reaction-environment-1 11235723>
  .                              .
  .                              .
  .                              .
```

Figure 6-17: An instance of ethane

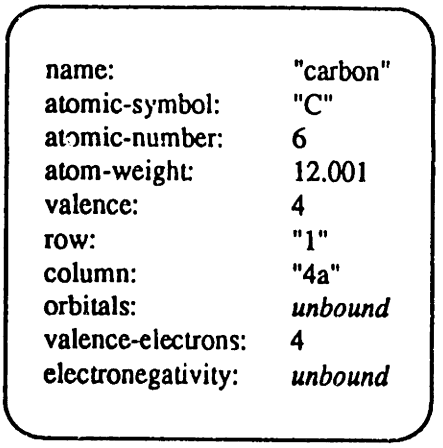


Figure 6-18: Description: database atom

weight, etc. Similarly, **atom** describes substrate dependent properties of an atom. These may include electronegativity, connecting bonds, hybridization, neighbor atoms, associated groups, etc. Figure 6-19 describes an instance of **atom** in the **abc** representing **ethane**. Figure 6-20 describes an instance of **bond** in the **abc** representing **ethane** and the linking and expansion of **db-bond**.

Similarly, Figure 6-21 shows the attributes that describe the group "methyl-1". The utility of **group** is that it makes explicit group properties. These properties: group-type, group-bonds, connecting-bonds, comprising-atoms are readily identified facilitating the development of known composite operators, **K**s, which are written around reaction centers (e.g. nitro or azo groups). Moreover, since groups are objects they can be manipulated directly, increasing the efficiency of reasoning processes which involve them (e.g. K_t or EQUIVALENT-MOLECULE-P[1]).

The representation described above makes important features of the **abc** explicit and directly accessible by providing abstractions of **abc**: meta-group abstraction, (i.e. groups constructed from groups), group abstraction, atom abstraction, and bond abstraction. For example, representations of **ethane** may include:

1. **ethyl-group-1** and **hydrogen-1** (ethyl is a meta-group abstraction of $-CH_3$ and $-CH_2-$).

2. **methyl-group-1** and **methyl-group-2** (group abstraction).

3. **atom-1** through **atom-8** (atom-abstraction)

4. **bond-1** through **bond-7** (bond-abstraction)

---

[1]EQUIVALENT-MOLECULE-P uses groups as one of many prescreening mechanisms (others include: **empirical-formula**, **abc-type**, and **bond-list**) prior to a node by node graph equivalency test.

identifier:	"C-T0764"
old-identifier:	NIL
free-valence:	0
diradical-p:	NIL
formal-charge:	0.0
electron-withdrawing-substituent-p:	NIL
connectivity-number:	4
hybridization:	"sp3"
conjugated-p:	NIL
p-orbitals:	NIL
open-approach-p:	T
parent-molecule:	#<ORGANIC-MOLECULE C2H6-T0779>
progenitors:	NIL
parent-groups:	(#<GROUP methyl-1> #<GROUP terminal-sp3-methylene-1> #<GROUP ethyl-1> #<GROUP ethyl-2>)
neighbor-atoms:	(#ATOM C-T0768> #<ATOM H-T0767> #ATOM H-T0766> #<ATOM H-T0765>)
neighbor-groups:	(#<GROUP methyl-2> #<GROUP terminal-sp3-methylene-2>)
bonds:	(#<BOND b-T0775> #<BOND b-T0774> #<BOND b-T0773> #<BOND b-T0772>)
DB-atom:	#<DB-ATOM 414001073>

Figure 6-19: Description: atom

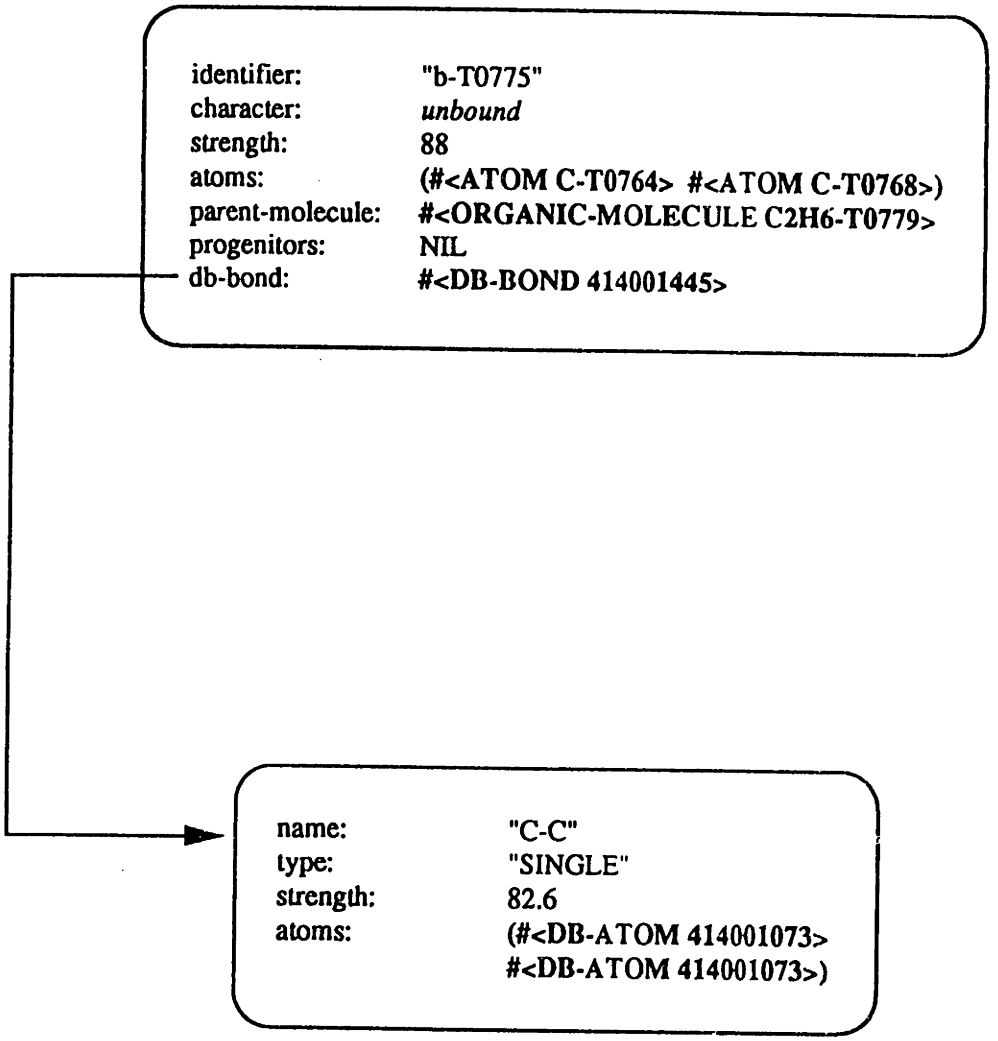


Figure 6-20: Bond description with database bond expansion

identifier:	"methyl-1"
string-identifier:	"- CH3"
type:	*unbound*
neighbor-groups:	(#<GROUP methyl-2> #<GROUP terminal-sp3-methylene-2> #<GROUP ethyl-1> #<GROUP ethyl-2>)
comprising-atoms:	(#ATOM C-T0764> #<ATOM H-T0767> #ATOM H-T0766> #<ATOM H-T0765>)
connecting-bonds:	#<BOND b-T0775>
connecting-atoms:	#<ATOM C-T0768>
group-weight:	15.035001
parent-molecule:	#<ORGANIC-MOLECULE C2H6-T0779>
DB-group:	#<DB-GROUP methyl>

Figure 6-21: Description: methyl group

By utilizing the knowledge contained in the different representational levels, the reasoning process involving chemical species and the utilities which transform them can be substantially enhanced.

The identification of various free radical pathways and underlying elementary reactions involved in the pyrolysis of ethane are evaluated by applying FIND-ALL-PATHWAYS to the substrates; and, associating with that call the designated **reaction-environment** and the group of composite operators to be used (e.g. **K**, $\mathbf{K}^*$, $\mathbf{K}_{\text{ab-initio}}$). The call to FIND-ALL-PATHWAYS is shown below:

(FIND-ALL-PATHWAYS :*substrates* (**ethane**) :*operators* $\mathbf{K}_{\text{free-radical}}$
:*override-environment* **reaction-environment**)

FIND-ALL-PATHWAYS begins by calling the composite operators that comprise the domain of free radical operations on the substrates specified by the keyword argument :*substrates*. These operators loop over the substrate list and evaluate properties specified by $\mathbf{K}_{\text{free-radical}}$. Homolytic dissociation of **ethane** is initiated when FEASIBLE-P, a compound predicate method of $\mathbf{K}_{\text{initiation}}$, ($\mathbf{K}_{\text{initiation}}$ is an operation of $\mathbf{K}_{\text{free-radical}}$) is evaluated on the sites identified by $\mathbf{K}_{\text{get-sites}}$. These sites are passed to FEASIBLE-P which searches the **reaction-environment** to establish potential radical forming processes: thermal cleavage, photochemical cleavage, and oxidation-reduction processes. Attributes describing **reaction-environment** establish thermal cleavage as a likely initiation mechanism[2]; photochemical cleavage and initiation by oxidation-reduction processes are eliminated as because their attribute value's are unbound or nil (i.e. non true):

[2]The keyword *:initiator-p* of FIND-ALL-PATHWAYS allows the user to override evaluation of initiation processes.

reaction-environment

temperature: 1200 K

wavelength: nil

surface-type: non-metal

Using this knowledge, $K_{initiation}$ constructs a list of potentially-cleavable-bonds by applying IDENTIFY-WEAKEST-BONDS to the bond representation of each substrate contained in the substrate-list (i.e. **ethane**). IDENTIFY-WEAKEST-BOND returns a list of bonds, arranged in increasing strength. This is shown below:

IDENTIFY-WEAKEST-BOND applied to **bond-abstraction** ⇒

((bond-1) (bond-2 bond-3 bond-4 bond-5 bond-6 bond-7))

where:

bond-1 = carbon-carbon single bond connecting **methyl-1** to **methyl-2**

bond-2 to **bond-4** = carbon-hydrogen single bond on **methyl-1**

bond-3 to **bond-7** = carbon-hydrogen single bond on **methyl-2**

Bond selection is based on relative strength. A default energy difference of 10 kcal/mol is used by IDENTIFY-WEAKEST-BOND but this value can be changed by the user. Bonds of equivalent strength are listed together.

For each system of reactants, IDENTIFY-WEAKEST-BOND identifies the weakest substrate **bond**, and compares it with others in the system to ensure minimal global bond strength. Selected **bonds** are then appended to potentially-cleavable-bonds. A set of top level[3] global gueues, denoted ***queue-name***, facilitate this process; as shown later, these queues become invaluable as new reactants are added to the environment.

[3] within the Lisp environment

The weakest system **bond** is identified by applying IDENTIFY-ABSOLUTE-WEAKEST-BOND to the bond representation of **ethane**. The value of **weakest-bond** is the carbon-carbon single bond connecting **methyl-1** to **methyl-2**. That is:

IDENTIFY-ABSOLUTE-WEAKEST-BOND

applied to potentially-cleavable-bonds $\Rightarrow$ **bond-1**

Once selected, it is and associated with the global queue **weakest-bond**. Specifically,

weakest-bond $\Leftarrow$ **bond-1**

This queue prevents $\mathbf{K}_{\text{initiation}}$ from processing the weakest bond of individual molecules when the energy difference between those bonds is greater than the default value. It also insures that the initiation process is focused on substrates which are most likely to cleave in **reaction-environment**. For example, in a complex molecule or set of species where one particular bond is substantially weaker than others in the system (e.g. a peroxy bond), knowledge of **weakest-bond** focuses the initiation process on that bond.

Once potentially-cleavable-bonds and **weakest-bond** become known to $\mathbf{K}_{\text{initiation}}$, FEASIBLE-P evaluates the energy in **reaction-environment** and assesses whether their is sufficient energy for oond cleavage; SUFFICIENT-THERMAL-ENERGY-P performs this evaluation. When FEASIBLE-P evaluates true, $\mathbf{K}_{\text{initiation}}$ calls K_t on the bond selected for cleavage. Application of bond cleavage operations, operators of $\mathbf{K}_{\text{ab-initio}}$, to the weakest bond of **ethane** produces a **microhomolysis-reaction**. Each **micro-reaction** has associated with it reactants, products, and reaction stoichiometry. In addition, the bond selected for cleavage is explicit as are various other properties of the **reaction** (see Figure 6-22). Methods of kinetic operator are used to identify these properties; they include: COLLECT-PRODUCTS, COMPUTE-STOICHIOMETRY, COMPUTE-BOND-FAVORABILITY, ASSESS-THERMODYNAMIC-FAVORABILITY, COMPUTE-EQUILIBRIUM-CONVERSION, etc.

reactants:	(#<organic-molecule C2H6-T0779>)
products:	(#<methyl-1> #<methyl-2>)
stoichiometry:	((#<C2H6> . -1) (#<methyl> . +2))
bond-cleaved:	#<bond-1>
heat-of-reaction:	88 kcal/mol
enabling-conditions:	#<K_f>
composing-tranformation-equations:	#<K_t>
reaction-environment:	#<REACTION-ENVIRONMENT-1>
rate-expression:	#<RATE-EXPRESSION-2>
equilibrium-constant:	#<EQUILIBRIUM-EXPRESSION-2>
context:	*unbound*
.	.
.	.
.	.

Figure 6-22: Description: elementary initiation reaction

Information pertinent to the transformation is associated with **micro-reaction** to make it easily accessible by methods and operations external to **micro-reaction**. Information essential to the external environment includes:

micro-homolysis-reaction

reactants: **(ethane)**

products: **(methyl-radical-1 methyl-radical-2)**

stoichiometry: **((ethane** . -1)

(methyl-radical-1 . +1) **(methyl-radical-2** . +1))

bond-cleaved: **bond-1**

environment: **reaction-environment**

Since multiple bonds may reside in potentially-cleavable-bonds, each possibly leading to a **micro-reaction**, $K_{initiation}$ creates a **global-reaction** which acts as a place holder for managing high level information: bond-queue and **micro-reactions**. **Global-reactions** maintain a record of this information and the association of a **micro-reaction** corresponding to a particular bond cleavage:

global-homolysis-reaction

bonds-to-be-cleaved: **(bond-1** ... **bond-*n*)**

weakest-bond: **(bond-1)**

micro-reaction: **(micro-homolysis-1** ... **micro-homolysis-*n*)**

K_t, the function responsible for performing a particular transformation, calls methods contained in $K_{ab\text{-}initio}$. Principle methods invoked by K_t, to affect homolytic bond cleavage of **ethane**, are GENERAL-SCISSION and CLEAVE-BOND. These methods often utilize helping functions, contained in various modeling elements (e.g. IDENTIFY-BOND-TYPE, HOMOLYTIC-P, IDENTIFY-CHARGE-DISTRIBUTION), to assist the transformation process. After a molecular bond has been selected for cleavage, it is passed to GENERAL-

SCISSION by $K_{ab\text{-}initio}$. Operations comprising GENERAL-SCISSION, identify the bonds parent-structure, perform the cleaving function, and manage fragments resulting from the cleavage process.

Fragment management is facilitated by ABSTRACT-GROUPING and ABSTRACT-ATOM-GROUPING. These methods partition an **abc** given a set of starting points (e.g. atoms). Normally, these points are specified by the end points (e.g. connecting atoms) of the reaction center identified by $K_{get\text{-}sites}$. In **ethane** disassociation, the reaction center end points are the two carbon atoms associated with the carbon-carbon single bond. As a consequence, the application of CLEAVE-BOND to **weakest-bond** (i.e. the element specified by **weakest bond**), two **abc**'s are identified. These are grouped by ABSTRACT-ATOM-GROUPING and become radicals (i.e **methyl-radical-1** and **methyl-radical-2** (see Figure 6-23)) upon instantiation by **abc**.

The **(abc)** instantiation process evaluates, updates, and classifies the **abc** as a **radical**. During instantiation, **abc** invokes MAP-OLD-ATOM-TO-NEW-ATOM: a method that maintains pointers within the chemical system so that atoms comprising a substrate know where they came from. This is accomplished using attribute's identifier and old-identifier. During the dissociation of **ethane** the value of **carbon-1** identifier's is "C-T0764" while its old-identifer ions [illegible]
database of user specification of the **abc**). However, the attribute value of old-identifier for **carbon-3**, the carbon comprising the new methyl radical, **methyl-radical-1**, reflects its progenitor **carbon-1**:

carbon-3		**carbon-1**	
identifier:	"C-T0790	identifier:	"C-T0764
old-identifier:	"C-T0764"	old-identifier:	nil
...	...	...	...

```
identifier:          "CH3-T0586"
name:                "methyl-radical"
atoms:               (#<ATOM C-T0780> #<ATOM H-T0781> . . .)
bonds:               (#<BOND b-T0784>   #<BOND b-T0785>
                      #<BOND b-T0786>)
equivalent-atoms:    ((#<ATOM H-T0781>  #<ATOM H-T0782>
equivalent-bonds:     #<ATOM H-T0783>) (#ATOM C-T0780>))
                     ((#<BOND b-T0784>    #<BOND b-T0785>
                      #<BOND b-T0786>))
groups:              (#<GROUP methyl-1>)
progenitors:         (#<MOLECULE C2H6-T0779>)
environment:         #<REACTION-ENVIRONMENT-1>
   .                    .
   .                    .
   .                    .
```

Figure 6-23: Description: methyl radical

Since the value of old-identifier describing **carbon-3** has the same value as **carbon-1**'s identifier value, these pointers enable construction of a substrate's complete history together with the operators which enabled each transformation.

A history trace is constructed by chaining through the values of the **abc** attribute progenitors. Similarly, the values of enabling-conditions, an attribute describing reaction, is traced to identify the various reactions comprising a particular pathway. These utilities, in combination, allow every attribute describing a modeling element (e.g. **abc**) to be logged and accessed, in the chronological order. This schema is one way in which CRL affords multifaceted and multilevel description of a modeling element throughout its history; and allows, for example, a multilevel description of a particular chemical species and the reaction pathways in which it participates, throughout its life cycle.

Once a composite operation has successfully been executed: FEASIBLE-P returns true and ab initio operations comprising the sequence of transformations contained in K_t return viable transformations; unique products identified by **reaction** are appended to the global queue denoted **potential-reactant-queue**. This queue contains a complete listing of potential reactants throughout the reaction cycle. Management of potential reactants in this manner allows the behavior of a system as it moves towards an equilibrium state, to be simulated.

Substrate screening prevents non-unique reactants from being appended to **potential-reactant-queue**. Similiarly, non-unique reactions are screened before being appended to **reaction-queue**. This prevents redundant entries in **potential-reactant-queue** and restricts a composite operation from being applied multiple times to the same reactant list. Figures 6-24 and 6-25 show **potential-reactant-queue** and **reaction-queue** respectively. These listings were obtained midway through the execution of FIND-ALL-

```
#<ORGANIC-RADICAL-MOLECULE C3H7-T1381>
#<ORGANIC-RADICAL-MOLECULE C4H9-T1323>
#<INORGANIC-MOLECULE H2-T1275>
#<RADICAL-MOLECULE H1-T0903>
#<ORGANIC-MOLECULE C2H4-T0901>
#<ORGANIC-RADICAL-MOLECULE C2H5-T0826>
#<ORGANIC-MOLECULE C1H4-T0812>
#<ORGANIC-RADICAL-MOLECULE C1H3-T0789>
#<ORGANIC-MOLECULE C2H6-T0779>
```

Figure 6-24: Description: *potential-reactant-queue*

```
((TERMINATION-REACTION #<ORGANIC-RADICAL-MOLECULE C3H7-T1381>
                       #<ORGANIC-RADICAL-MOLECULE C4H9-T1323> ))
 (PROPAGATION-REACTION #<ORGANIC-RADICAL-MOLECULE C4H9-T1323>
                       #<ORGANIC-MOLECULE C2H4-T0901>)
 (PROPAGATION-REACTION #<ORGANIC-RADICAL-MOLECULE C3H7-T1381>
                       #<ORGANIC-MOLECULE C2H4-T0901>)
 (PROPAGATION-REACTION #<ORGANIC-RADICAL-MOLECULE C3H7-T1381>
                       #<ORGANIC-RADICAL-MOLECULE C4H9-T1323>)
 (TERMINATION-REACTION #<RADICAL-MOLECULE H1-T0903>
                       #<ORGANIC-RADICAL-MOLECULE C1H3-T0789>)
 (TERMINATION-REACTION #<RADICAL-MOLECULE H1-T0903>
                       #<ORGANIC-RADICAL-MOLECULE C2H5-T0826>)
 (PROPAGATION-REACTION #<ORGANIC-MOLECULE C2H4-T0901>
                       #<ORGANIC-RADICAL-MOLECULE C1H3-T0789>)
 (PROPAGATION-REACTION #<ORGANIC-MOLECULE C2H4-T0901>
                       #<ORGANIC-RADICAL-MOLECULE C2H5-T0826>)
 (PROPAGATION-REACTION #<RADICAL-MOLECULE H1-T0903>
                       #<ORGANIC-MOLECULE C2H6-T0779>)
 (PROPAGATION-REACTION #<RADICAL-MOLECULE H1-T0903>
                       #<ORGANIC-MOLECULE C2H4-T0901>)
 (TERMINATION-REACTION #<ORGANIC-RADICAL-MOLECULE C2H5-T0826>
                       #<ORGANIC-RADICAL-MOLECULE C1H3-T0789>)
 (PROPAGATION-REACTION #<ORGANIC-RADICAL-MOLECULE C2H5-T0826>
                       #<ORGANIC-MOLECULE C1H4-T0812>
 (PROPAGATION-REACTION #<ORGANIC-RADICAL-MOLECULE C1H3-T0789>
                       #<ORGANIC-MOLECULE C2H6-T0779>)
 (INITIATION-REACTION  #<ORGANIC-MOLECULE C2H6-T0779>)
```

Figure 6-25: Description: *reaction-queue*

PATHWAYS applied to the substrate **ethane** in a reaction environment of 1200K; composite operations were limited to free radical operators during the evaluation of FIND-ALL-PATHWAYS.

Continued application of composite operations, invoked by FIND-ALL-PATHWAYS, on the substrates contained in **potential-reaction-queue** identifies additional **transfer, propagation,** and **termination reactions.** By tracing the history of these **reactions, free-radical-reactions** (i.e. free radical pathways) constructs a **free-radical-reaction.** Attributes describing a **free-radical-reaction** are shown in Figure 6-26. Methods of **free-radical-reaction** provide the utility for searching through a set of chemical species, using the value of the progenitors attribute and MOLECULE-EQUIVALENT-P,[4] to identify substrate loops within the propagation reaction network. Propagation reactions are then identified and the attribute value established.

However, due to the myriad of products which are capable of being formed in termination sequences, application of $K_{free\text{-}radical}$ to substrates in the **potential-reactant-queue** may not terminate. This potential exists because new reactants, resulting from **coupling-reactions** and **combination-reactions,** are constantly appended to **potential-reactant-queue**. For example, two **ethyl-radicals** can combine to form **butane;** hydrogen abstraction of **butane** can form **butyl** which may combine to form **octane.** This cycle can, in principle, continue indefinitely. To prevent such cycles, substrates are removed from **potential-reactant-queue** when they have been consumed, or when a homologous series of a reactant has been previously examined. The latter is accomplished using HOMOLOGOUS-SERIES-P, with the register function responsible for appending potential reactants to **potential-reactant-queue**.

---

[4] A predicate function which establishes equivalency among abc's.

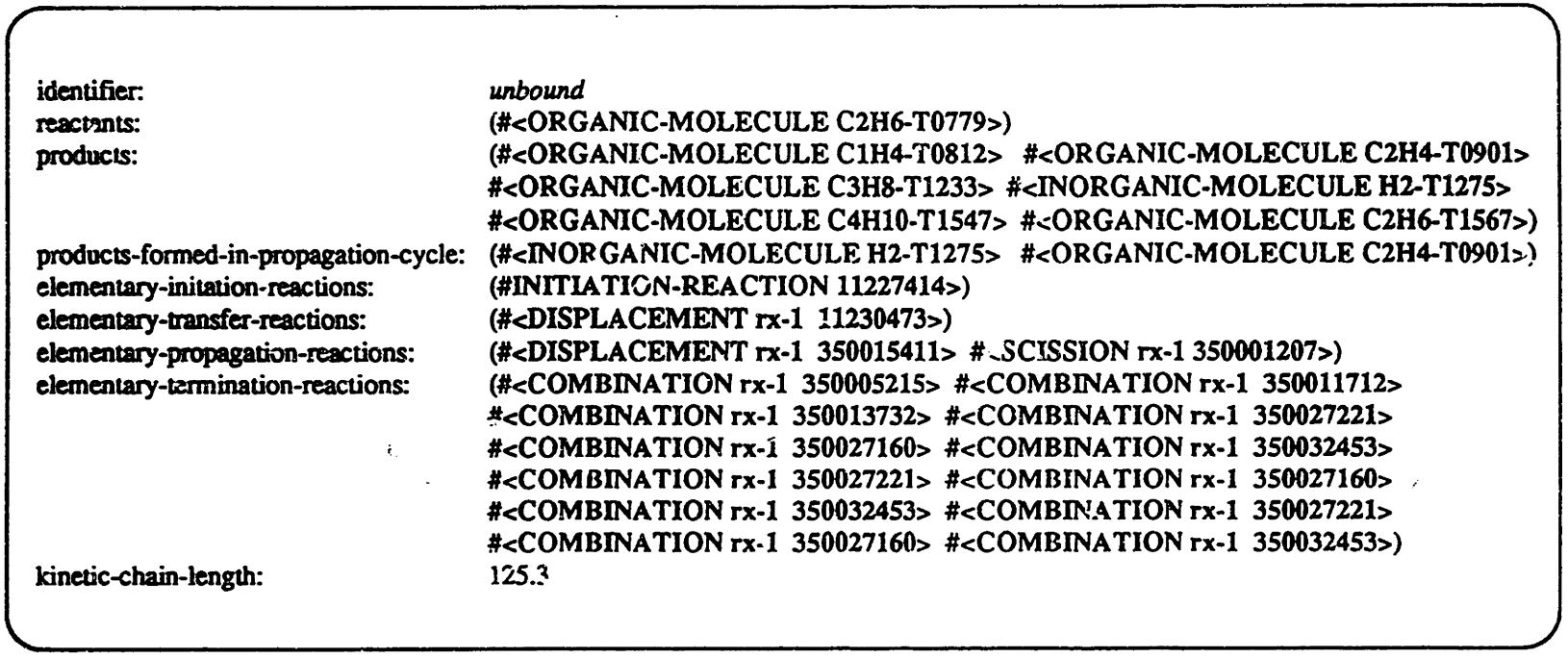


Figure 6-26: Description: free-radical-reaction

Pathways of the overall reaction sequence are constructed using **free-radical-reactions**. Information associated with the individual **reactions**: **initiation-reaction**, **transfer-reaction**, **propagation-reaction**, **branching-reaction**, and **termination-reaction**; facilitate this task. With this information, **free-radical-reactiond** is able to construct individual free radical pathways. Figure 6-27 shows the attributes of **free-radical-reactions**. The attribute generation-steps, describing **free-radical-reactions**, specifies the number of steps or loops required to establish quiescence in the global queues.

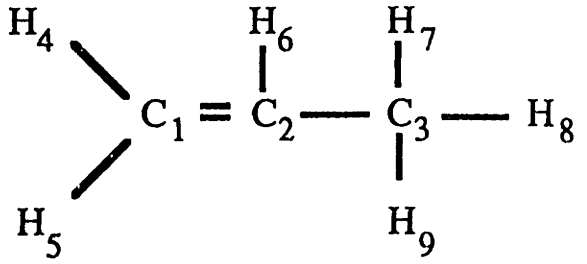
6.6.2 Case Study 2: chlorination of propylene

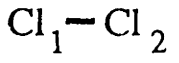
In this illustration we will focus on the chlorination of propylene. Specifically we will investigate alternative pathways for the chlorination process and how CRL manages the various reactions which result from alternative reaction centers. Moreover, we will highlight how evaluations are made when alternative reaction centers reside in a set of reactants.

We begin, as before, by initializing the **abcs** which represent a set of reactants under investigation. Since neither compound, propylene or chlorine, contains stereo centers, initialization to form the object graph representation is accomplished directly from the bond electron matrix shown in Figure 6-28. Transformation of the bond electron matrices to the **abcs** is accomplished by parsing the bond electron matrix using MAKE-GRAPH-FROM-SYMMETRIC-CONNECTIVITY-MATRIX. Since free valence electrons are represented in the be-matrix representation (see the be-matrix for chlorine in Figure 6-28), MAKE-GRAPH-FROM-SYMMETRIC-CONNECTIVITY-MATRIX is able to identify the types of **bonds** associated with each **atom**. The object graphs or **abcs**, representing these reactants, are shown in Figures 6-29 and 6-30; therein objects are denoted with a # prefix.

identifier:	*unbound*
initial -substrates:	(#<ORGANIC-MOLECULE C2H6-T0779>)
cyclic-products:	(#<INORGANIC-MOLECULE H2-T1275> #ORGANIC-MOLECULE C2H4-T0901>)
cyclic-reactions:	(#<FREE-RADICAL-REACTION 35003651>)
cyclic-operators:	((#<DISPLACEMENT rx-1 350015411> #<SCISSION rx-1 350001207>))
generation-steps:	((#<FREE-RADICAL-GENERATION-STEP 350016621> #<FREE-RADICAL-GENERATION-STEP 350001336> #<FREE-RADICAL-GENERATION-STEP 11233575>))

Figure 6-27: Description: free-radical-reactions





	1	2	3	4	5	6	7	8	9
1	0	1	0	1	1	0	0	0	0
2	1	0	1	0	0	1	0	0	0
3	0	1	0	0	0	0	1	1	1
4	1	0	0	0	0	0	0	0	1
5	1	0	0	0	0	0	0	0	0
6	0	1	0	0	0	0	0	0	0
7	0	0	1	0	0	0	0	0	0
8	0	0	1	0	0	0	0	0	0
9	0	0	1	0	0	0	0	0	0

	1	2
1	6	1
2	1	6

Figure 6-28: be-matrix of propylene and chlorine

```
identifier:                  "C3H6-T0617"
name:                        "propylene"
atoms:                       (#<ATOM C-T0600> #<ATOM H-T0601>
                              #<ATOM H-T0602> #<ATOM H-T0603>
                              #<ATOM C-T0604> #<ATOM H-T0605>
                              #<ATOM C-T0606> #<ATOM H-T0607> #<ATOM H-T0608>)
bonds:                       (#<BOND b-T0609>  #<BOND b-T0610>
                              #<BOND b-T0611>  #<BOND b-T0612>
                              #<BOND b-T0613>  #<BOND b-T0614>
                              #<BOND b-T0615>  #<BOND b-T0616>)
empirical-formula:           ((#<DB-ATOM 360002120> . 6) (#<DB-ATOM 360002161> . 3))
empirical-formula-string:    "C3H6"
molecular-weight:            42.080997
charge:                      0.0
terminal-skeleton-atoms:     (#<ATOM C-T0600>  #<ATOM C-T0606>)
equivalent-atoms:            ((#<ATOM C-T0600>)
                              (#<ATOM H-T0603> #<ATOM H-T0601> #<ATOM H-T0602>)
                              (#<ATOM C-T0604>)
                              (#<ATOM C-T0605>)
                              (#<ATOM C-T0606>)
                              (#<ATOM H-T0608> #<ATOM H-T0607>))
equivalent-bonds:            ((#<BOND b-T0611> #<BOND b-T0609> #<BOND b-T0610>)
                              (#<BOND b-T0612>)
                              (#<BOND b-T0613>) (#<BOND b-T0614>)
                              (#<BOND b-T0616> #<BOND b-T0615>))
weakest-bond:                (#<BOND b-T0612>)
weakest-bond-strength:       82.6
weakest-bond-strength-ratio: 1.0
ordered-eq-bonds:            ((#<BOND b-T0612>)
                              (#<BOND b-T0611> #<BOND b-T0609> #<BOND b-T0610>)
                              (#<BOND b-T0613>)
                              (#<BOND b-T0616> #<BOND b-T0615>)
                              (#<BOND b-T0614>))
groups:                      (#<GROUP methyl-1> #<GROUP terminal-sp3-methylene-1>
                              #<GROUP terminal-sp2-methylene-1> #<GROUP vinyl-1>
                              #<GROUP terminal-allyl-1> #<GROUP ethnynyl-1)
                               #<GROUP terminal-sp3-methylene-2>)
group-bonds:                 (#<BOND-b-T0609> #<BOND b-T0614>
                              #<BOND b-T0611> #<BOND b-T0616>
                              #<BOND b-T0615> #<BOND b-T0612>)
methyl-carbon:               NIL
primary-carbons:             (#<ATOM C-T0600> #<ATOM C-T0606>)
secondary-carbons:           (#<ATOM C-T0604>)
tertiary-carbons:            NIL
terminal-carbons:            (#<ATOM C-T0600> #<ATOM C-T0606>)
backbones:                   (#<ATOM C-T0600> #<ATOM C-T0604> #<ATOM C-T0606>)
backbone-length:             3
progenitor:                  NIL
environment:                 NIL
   .
   .
   .
```

Figure 6-29: An instance of propylene

identifier:	"Cl2-T0658"
name:	"chlorine"
atoms:	(#<ATOM Cl-T0655> #<ATOM Cl-T0656>)
bonds:	(#<BOND b-T0657> #<BOND b-T0777> #<BOND b-T0778>)
empirical-formula:	((#<DB-ATOM 414000575> . 6) (#<DB-ATOM 414001073> . 2))
empirical-formula-string:	"Cl2"
molecular-weight:	70.906
charge:	0.0
terminal-skeleton-atoms:	NIL
equivalent-atoms:	((#<ATOM Cl-T0656> #<ATOM Cl-T0655>))
equivalent-bonds:	(#<BOND b-T0657>)
weakest-bond:	(#<BOND b-T0657>)
weakest-bond-strength:	57.83939
weakest-bond-strength-ratio:	0.0
ordered-eq-bonds:	((#<BOND b-T0657>))
groups:	NIL
group-bonds:	NIL
meta-groups:	unbound
progenitor:	NIL
environment:	NIL
D-electrons:	0

Figure 6-30: An instance of chlorine

Using the methods of CRL thermochemical energies of a **bond** can be evaluated. Bond energy assessment is achieved in one of three ways: Firstly, database lookup of thermochemical bond energies for single bonds and multiple bonds. These energies do not account for neighbor effects. Secondly, database lookup of bond dissociation energies for model organic molecules (R'-R''). Generally, R' entries are for common organic fragments such as:

$-CH_3$	t-Bu-
$-C_2H_5$	$-C_6H_5$
$n-C_3H_7-$	$C_6H_5CH_2-$
$i-C_3H_7-$	$-COCH_3$

R'' normally includes entries for single atomic constituents or small groups. These include:

-H	-Br	$-C(CH_3)_3$
-F	-OH	$-OCH_3$
-Cl	$-NH_3$	$-CH_3$

Thirdly, semi-empirical bond energy estimation. This is accomplished by creating a model of the bond center of interest and comparing this model with analogous centers contained in the databases. Abstract modeling of the bond center is achieved using **groups** and and **meta-groups** of the **abc**. Using the **groups** and connecting entities (e.g. **atoms, groups, meta-groups**) methods of CRL abstract the bond and its chemical environment so that it can use the knowledge contained in the databases, whenever possible. These databases contain in excess of 1000 entries for thermochemical bond dissociations.

The identification of various free radical pathways and underlying elementary reactions involved in the chlorination of propylene are evaluated by applying FIND-ALL-PATHWAYS to the substrates; and, associating with that call the designated **reaction-environment** and the group of composite operators of choice. The call to FIND-ALL-PATHWAYS is shown below:

Using this knowledge, $K_{initiation}$ constructs a list of potentially-cleavable-bonds by applying IDENTIFY-WEAKEST-BONDS to the bond representation of each substrate contained in the substrate-list (i.e. **propylene** and **chlorine**). IDENTIFY-WEAKEST-BOND returns a nested list of bonds, arranged in increasing strength: bonds of equivalent strength are listed together. This is shown below:

IDENTIFY-WEAKEST-BOND applied to **bond-abstraction** of **chlorine** ⇒

(bond-1)

where:

bond-1 = chlorine-chlorine single bond connecting **chlorine-1** to **chlorine-2**

and,

IDENTIFY-WEAKEST-BOND applied to **bond-abstraction** of **propylene** ⇒

((bond-1) (bond-2) (bond-3))

(bond-4)

(bond-7)

((bond-5 "sp2-carbon-hydrogen") **(bond-6** "sp2-carbon-hydrogen"))

(bond-8 "sp2-carbon-sp2-carbon"))

where:

bond-1 to **bond-3** = carbon-hydrogen single bond of **methyl-1**

bond-4 = carbon-carbon single bond connecting **methyl-1** to **ethylene-1**

bond-7 = secondary carbon-hydrogen single bond of **ethylene-1**

bond-5 and **bond-6** = terminal carbon-hydrogen single bond of **ethylene-1**

bond-8 = carbon-carbon double bond of **ethylene-1**

For each system of reactants, IDENTIFY-WEAKEST-BOND identifies the weakest substrate **bond**, and compares it with others in the system to ensure minimal global bond strength. Selected **bonds** are then appended to potentially-cleavable-bonds. The weakest system **bond** is identified by applying IDENTIFY-ABSOLUTE-WEAKEST-BOND to potentially-cleavable-bonds. The value of **weakest-bond** is the chlorine-chlorine single bond connecting **chlorine-1** to **chlorine-2**. That is:

IDENTIFY-ABSOLUTE-WEAKEST-BOND

applied to potentially-cleavable-bonds ⇒ <**bond-1** "Cl-Cl">

Once selected, this **bond** is associated with the global queue **weakest-bond**:

weakest-bond ⇐ <**bond-1** "Cl-Cl">

This queue prevents $K_{initiation}$ from processing the weakest bond of **propylene** when the energy difference between the weakest bond of chlorine is greater than the default value of 10 kcal/mol. In this circumstance, cleavage of chlorine is favored over the cleavage of the allylic hydrogen by 30 kcal/mol. Identification of **weakest-bond** and its bond strength ensures that cleavage will be focused on the weakest bonds of the system as new substrates are added to the reaction system.

After potentially-cleavable-bonds and **weakest-bond** become known to $K_{initiation}$, FEASIBLE-P evaluates the energy in **reaction-environment**. FEASIBLE-P calls SUFFICIENT-THERMAL-ENERGY-P to assess whether <**bond-1** "Cl-Cl"> will cleave. When FEASIBLE-P evaluates true, $K_{initiation}$ calls K_t on <**bond-1** "Cl-Cl">. Application of bond cleavage operations to the weakest bond of **chlorine** produces the

micro-homolysis-reaction shown below; a more complete description is shown in Figure 6-31

micro-homolysis-reaction

reactants:	**(chlorine)**
products:	**(chloro-radical-1 chloro-radical-2)**
stoichiometry:	**((chlorine . -1)**
	(chloro-radical-1 . +1) (chloro-radical-2 . +1))
bond-cleaved:	**<bond-1** "Cl-Cl">
environment:	**reaction-environment**

Once $K_{initiation}$ has been successfully executed, forming **micro-homolysis-reaction**, unique products identified by **micro-homolysis-reaction** are appended to the global queue, **potential-reactant-queue**. This queue provides a window into the reaction system via the potential reactants.

Substrates contained in **potential-reactant-queue** become potential reactants in future calls on the operations comprising FIND-ALL-PATHWAYS. These operators loop over the substrates of **potential-reactant-queue** and evaluate properties specified by $K_{free-radical}$. Characterization of **chloro-radical-1** by **chemical-behavior** restricts the number of composite operations applied to **chloro-radical-1**. This is due to the reactivity associated with **singly-occupied-molecular-orbital** which pilots the search of $K_{get-sites}$.

Five potential reaction centers are identified by $K_{get-sites}$ for the attack of **chloro-radical-1** on **propylene**. These **sites** are passes to FEASIBLE-P which calls for an initial evaluation of the potential reaction center pairs. Initial evaluation of these pairs suggests the following possible reactions if K_t was to be executed on them:

reactants:	(#<inorganic-molecule Cl2-T0658>)
products:	(#<chloro-1> #<chloro-2>)
stoichiometry:	(#<Cl2> . -1) (#<chloro> . +2))
bond-cleaved:	#<BOND b-T0657>
heat-of-reaction:	58 kcal/mol
enabling-conditions:	#<K_f>
composing-transformation-equations:	#<K_t>
reaction environment:	#<REACTION-ENVIRONMENT>
rate-expression:	#<RATE-EXPRESSION-4>
equilibrium-constant:	#<EQUILIBRIUM-EXPRESSION-4>
context:	*unbound*
.	.
.	.
.	.

Figure 6-31: Description: elementary homolysis reaction

1. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow CH_2=CH-{}^{\bullet}CH_2 + HCl$

2. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow CH_2=C^{\bullet}-CH_3 + HCl$

3. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow {}^{\bullet}CH=CH-CH_3 + HCl$

4. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow CH_2Cl-{}^{\bullet}CH-CH_3$

5. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow {}^{\bullet}CH_2-CHCl-CH_3$

The radicals formed from these reactions are a result of two principle mechanisms: hydrogen abstraction by chlorine radical, and chlorine addition to the pi system. FEASIBLE-P screens this potential reaction list by calling ASSESS-THERMODYNAMIC-FAVORABILITY on the pairs of reaction centers. It then passes the most favorable of those reaction centers to K_t. The number of reaction center pairs passes to K_t is based on the energy difference (e.g. kcal/mol) specified by the user; in the absence of such specification CRL uses a default value of 10 kcal/mol for similar classes of reactions.

Initial screening results in one abstraction and two addition reaction centers being passed to K_t. These centers represent the following reactions:

1. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow CH_2=CH-{}^{\bullet}CH_2 + HCl$

2. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow CH_2Cl-{}^{\bullet}CH-CH_3$

3. $CH_2=CH-CH_3 + Cl^{\bullet} \rightarrow {}^{\bullet}CH_2-CHCl-CH_3$

$K_{propagation}$ then calls K_t on the reaction center pairs selected. Application of various $K_{ab\text{-}initio}$ operations to the reaction centers identified by $K_{get\text{-}sites}$, and called by K_t, produces a series of **micro-addition-reactions** and **micro-displacement-reactions**. Each **micro-reaction** has associated with it reactants, products, and reaction stoichoimetry. In addition, the reaction centers selected are made explicit. This

information is essential to the external environment and therefore has been associated with **micro-reaction** to make it easily accessible by other operations of CRL. Essential information concerning the primary **micro-displacement-reaction** identified, is shown below; a more complete description is shown in Figure 6-32:

micro-displacement-reaction

reactants:	**(propylene chloro-radical-1)**
products:	**(propyl-radical-1 inorganic-molecule-1)**
stoichiometry:	**((propylene . -1) (chloro-radical . -1)**
	(propylene-radical-1 . +1)
	(inorganic-molecule-1 . +1))
reaction-center:	**carbon-3**
environment:	**reaction-environment**

Key **micro-addition-reactions**, which lead to the competitive formation of 1, 2-dichloropropane, are shown below as well:

micro-addition-reaction

reactants:	**(propylene chloro-radical-1)**
products:	**(1-chloro-2-propyl-radical-1)**
stoichiometry:	**((propylene . -1) (chloro-radical . -1)**
	(1-chloro-2-propyl-radical-1 . +1))
reaction-center:	**carbon-1**
environment:	**reaction-environment**

reactants:	(#<organic-molecule C3H6-T0617>)
products:	(#<organic-radical C3H5-T0683> #<inorganic-molecule HCl-T0682>)
stoichiometry:	((#<C3H6> . -1) (#<Cl> . -1) (#<Cl> . 1) (#<C3H5> . 1))
reaction-centers:	(#<ATOM C-T0606> #<ATOM Cl-T0655>)
heat-of-reaction:	-15 kcal/mol
enabling-conditions:	#<K_f>
composing-transformation-equations:	#<K_t>
reaction-environment:	#<REACTION-ENVIRONMENT>
rate-expression:	#<RATE-EXPRESSION-9>
equilibrium-constant:	#<EQUILIBRIUM-EXPRESSION-10>
context:	*unbound*
.	.
.	.
.	.

Figure 6-32: Description: elementary displacement reaction

micro-addition-reaction

reactants: **(propylene chloro-radical-1)**

products: **(2-chloro-1-propyl-radical-2)**

stoichiometry: **((propylene . -1) (chloro-radical . -1)**

(2-chloro-1-propyl-radical-1 . +1))

reaction-center: **carbon-2**

environment: **reaction-environment**

Depending on the chemistry involved, multiple reaction centers may reside in the substrates contained in **potential-reactant-queue**. Since each center may lead to a separate **micro-reaction**, as shown above, $K_{propagation}$ creates a **global-reaction** for managing high level information: reaction-center-queue and **micro-reactions**. **Global-reaction** maintains a record of the centers to be processed and an association of a **micro-reaction** resulting from the processing of those centers:

global-propagation-reaction

reaction-centers: **(reaction-center-1...reaction-center-*n*)**

micro-reaction: **(micro-addition-reaction-1**

... micro-displacement-reaction-*n*)

Methods embedded in $K_{propagation}$ provide the means for evaluating the probability of one **micro-reaction** over another. These methods draw on knowledge derived by and contained in **micro-reactions** (e.g. thermodynamic favorability); as well as, knowledge derived by **cb** which allows relative reactivity and steric effects to be evaluated.

The unique products identified by **micro-reaction** are then appended to the global queue **potential-reactant-queue**. Continued application of the composite operations, invoked by FIND-ALL-PATHWAYS, on the substrates contained in **potential-reactant-queue** identifies additional **transfer, propagation, and termination reactions.** By tracing the

history of these reactions, **free-radical-reaction** constructs a free radical pathway. Attributes describing **free-radical-reactions**, for the chlorination of propylene, are shown in Figure 6-33. Expansion of **free-radical-reaction-1**, identified by **free-radical-reactions**, is shown in Figure 6-34. Pathways describing **free-radical-reaction-1**, **free-radical-reaction-2**, and **free-radical-reaction-3** are shown in Figures 6-35, 6-36, and 6-37, respectively.

6.6.3 Case Study 3: oxidation of butane

In this illustration we will focus on the oxidation of butane. Specifically we will show how CRL constructs reaction pathways from a given set of elementary reactions and how it identified competing pathways. We will also highlight the use of the context in pathway generation.

We begin by initializing the **abcs** which represent the set of reactants under investigation. In this illustration we introduce an initiator into the set of reactants so that low temperatures can be maintained. Transformation of the bond electron matrices to the **abcs** is accomplished by parsing the bond electron matrix using MAKE-GRAPH-FROM-SYMMETRIC-CONNECTIVITY-MATRIX. The object graphs for the reactants excluding the initiator are shown in Figures 6-38 and 6-39; therein objects are denoted with a # prefix.

Identification of various free radical pathways and underlying elementary reactions involved in the oxidation of butane are evaluated by applying FIND-ALL-PATHWAYS to the substrates; and, associating with that call the designated **reaction-environment** and the group of composite operators of choice. The call to FIND-ALL-PATHWAYS is shown below:

identifier:	*unbound*
initial-substrates:	(#<ORGANIC MOLECULE C3H6-T0617>)
	(#<INORGANIC MOLECULE Cl2-T0658>)
cyclic-products:	(#<INORGANIC MOLECULE HCl-T0682>)
	(#<ORGANIC MOLECULE C3H5Cl-T0661>)
	(#<ORGANIC MOLECULE C3H6Cl2-T0403>)
	(#<ORGANIC MOLECULE C3H6Cl2-T0403>)
	(#<ORGANIC MOLECULE C3H6Cl2-T0404>)
cyclic-reactions:	(#<FREE-RADICAL-REACTION-350063531>)
	(#<FREE-RADICAL-REACTION-350063532>)
	(#<FREE-RADICAL-REACTION-350063533>)
	(#<FREE-RADICAL-REACTION-350063534>)
	.
	.
	.
cyclic-operators:	(#<ADDITION 350014309>)
	(#<DISPLACEMEN 350014512>)
	(#<ADDITION 350014411>)
	(#<DISPLACEMENT 350017201>)
	.
	.
	.
cyclic-steps:	(#<FREE-RADICAL-GENERATION-STEP 350012597>)
.	(#<FREE-RADICAL-GENERATION-STEP 250012599>)
.	.
.	.
	.

Figure 6-33: Propylene chlorination: free-radical-reactions description

identifier:	"Free-Radical-Reaction-1 350063531"
reactants:	(#<ORGANIC MOLECULE C3H6-T0617> #<INORGANIC MOLECULE Cl2-T0658>)
products:	(#<ORGANIC MOLECULE C3H5Cl-T0661>) (#<INORGANIC MOLECULE Cl2-T0690> #<INORGANIC MOLECULE HCl-T0682>) (#<ORGANIC MOLECULE C6H10-T0694>)
products-formed-in-propagation-cycle:	(#<INORGANIC MOLECULE HCl-T0682> #<ORGANIC MOLECULE C3H5Cl-T0661>)
elementary-initation-reactions:	(#<INITIATION REACTION 11227114>)
elementary-propagation-reactions	(#<DISPLACEMENT 350014512> #<DISPLACEMENT 350017201>)
elementary-termination-reactions:	(#<COMBINATION 350015251> #<COMBINATION 350071121>) (#<COMBINATION 350017323> #<COMBINATION 350072212>)
kinetic-chain-length:	8128

Figure 6-34: Description: free-radical-reaction-1

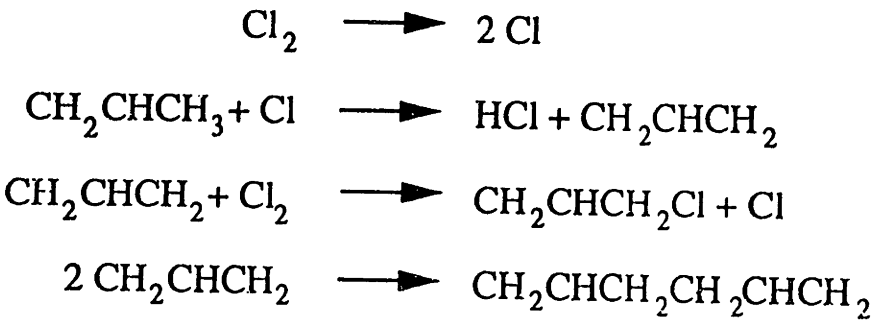


Figure 6-35: Pathway Descriptions: free-radical-reaction-1

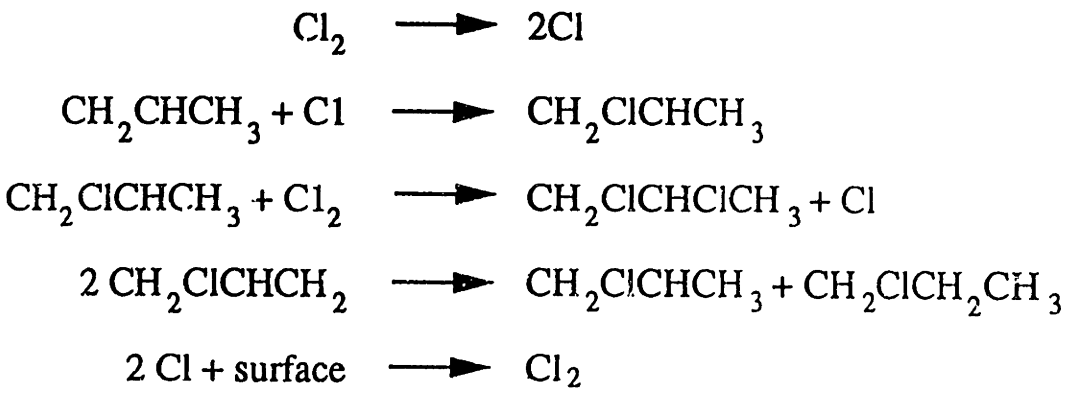


Figure 6-36: Pathway Descriptions: free-radical-reaction-2

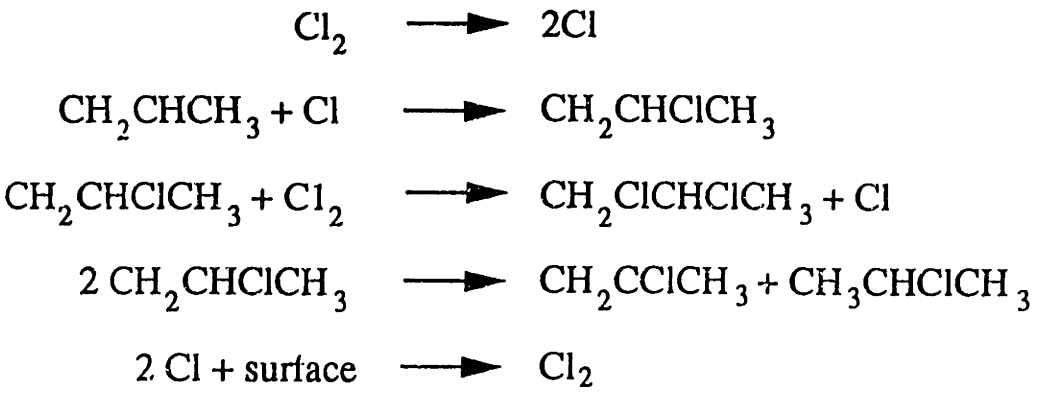


Figure 6-37: Pathway Descriptions: free-radical-reaction-3

identifier:	"C4H10-T0702"
name:	"Butane"
atoms:	(#<ATOM C-T0675> #<ATOM H-T0676> #<ATOM H-T0677> #<ATOM H-T0678> #<ATOM C-T0679> #<ATOM H-T0680> #<ATOM H-T0681> #<ATOM C-T0682> #<ATOM H-T0683> #<ATOM H-T0684> #<ATOM C-T0685> #<ATOM H-T0686> #<ATOM H-T0687> #<ATOM C-T0688>)
bonds:	(#<BOND b-T0689> #<BOND b-T0690> #<BOND b-T0691> #<BOND b-T0692> #<BOND b-T0693> #<BOND b-T0694> #<BOND b-T0695> #<BOND b-T0696> #<BOND b-T0697> #<BOND b-T0698> #<BOND b-T0699> #<BOND b-T0700> #<BOND b-T0701>)
empirical-formula:	((#<DB-ATOM 415164516> . 10) (#<DB-ATOM 415156345> . 4))
empirical-formula-string:	"C4H10"
molecular-weight:	58.124
charge:	0.0
terminal-skeleton-atoms:	(#<ATOM C-T0675> #<ATOM C-T0685>)
equivalent-atoms:	((#<ATOM C-T0682> #<ATOM C-T0679>) (#<ATOM H-T0684> #<ATOM H-T0680> #<ATOM H-T0681> #<ATOM H-T0683>) (#<ATOM C-T0685> #<ATOM C-T0675>) (#<ATOM H-T0688> #<ATOM H-T0676> #<ATOM H-T0677> #<ATOM H-T0678> #<ATOM H-T0686> #<ATOM H-T0687>))
equivalent-bonds:	((#<BOND b-T0695>) (#<BOND b-T0697> #<BOND b-T0693> #<BOND b-T0694> #<BOND b-T0696>) (#<BOND b-T0698> #<BOND b-T0692>) (#<BOND b-T0701> #<BOND b-T0689> #<BOND b-T0690> #<BOND b-T0691> #<BOND b-T0699> #<BOND b-T0700>))
weakest-bond:	(#<BOND b-T0698>)
weakest-bond-strength:	82.6
weakest-bond-strength-ratio:	1.0
ordered-eq-bonds:	((#<BOND b-T0695>) (#<BOND b-T0698> #<BOND b-T0692>) (#<BOND b-T0697> #<BOND b-T0693> #<BOND b-T0694> #<BOND b-T0696>) (#<BOND b-T0701> #<BOND b-T0689> #<BOND b-T0690> #<BOND b-T0691> #<BOND b-T0699> #<BOND b-T0700>))
groups:	(#<GROUP methyl-1> #<GROUP methyl-2> #<GROUP internal-sp3-methylene-1> #<GROUP internal-sp3-methylene-2> #<GROUP general-internal-sp3-methylene-1> #<GROUP general-internal-sp3-methylene-2> #<GROUP terminal-sp3-methylene-1> #<GROUP terminal-sp3-methylene-2>))
group-bonds:	(#<BOND-b-T0693> #<BOND b-T0696> #<BOND b-T0692> #<BOND b-T0689> #<BOND b-T0699> #<BOND b-T0698> #<BOND b-T0695>)
methyl-carbon:	NIL
primary-carbons:	(#<ATOM C-T0675> #<ATOM C-T0685>)
secondary-carbons:	(#<ATOM C-T0679> #<ATOM C-T0682>)
tertiary-carbons:	NIL
terminal-carbons:	(#<ATOM C-T0675> #<ATOM C-T0685>)
backbones:	(#<ATOM C-T0675> #<ATOM C-T0685>) (#<ATOM C-T0682> #<ATOM C-T0685>)
backbone-length:	4
progenitor:	NIL
environment:	#<reaction-environment-3>

Figure 6-38: An instance of butane

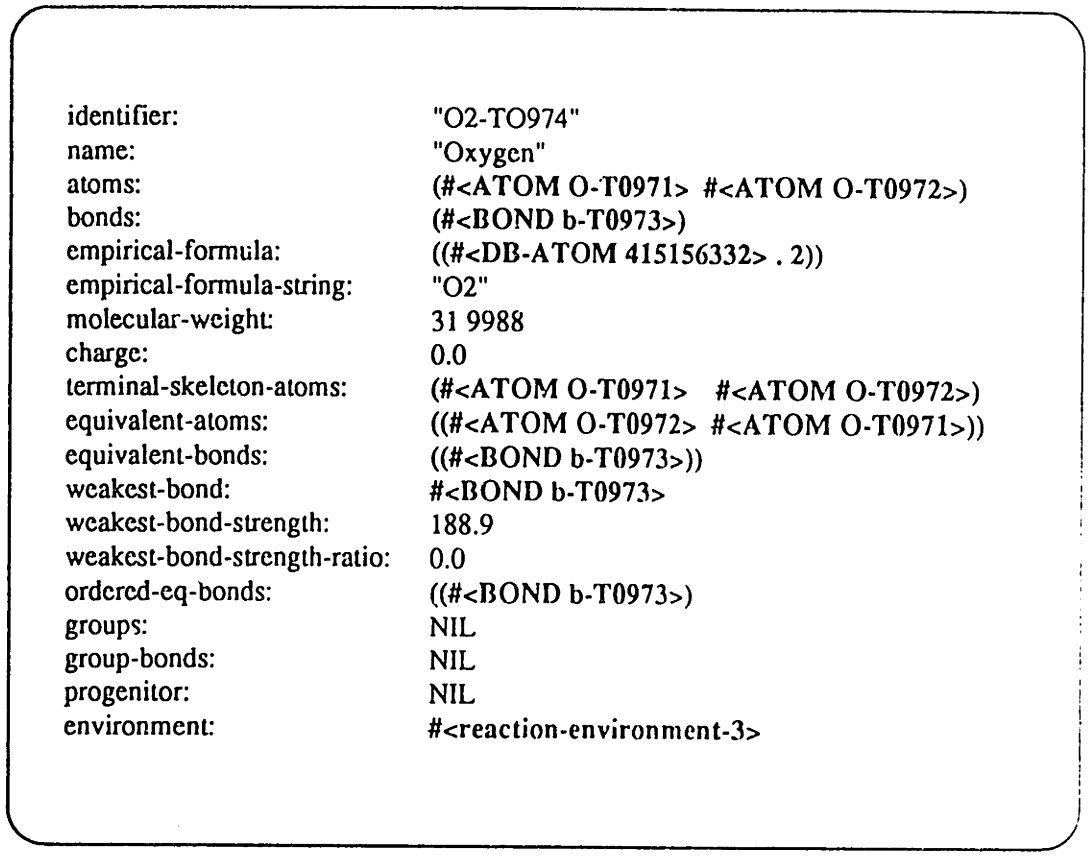


Figure 6-39: An instance of oxygen

(FIND-ALL-PATHWAYS :*substrates* (**butane oxygen**
sec-butyl-hydroperoxide)
:*operators* $\mathbf{K}_{\text{free-radical}}$
:*override-environment* **reaction-environment**)

Alternatively the call to FIND-ALL-PATHWAYS could exclude the use of the initiator from the substrate list by setting the keyword *initiator-p*to true. In this situation, the call would be as follows:

(FIND-ALL-PATHWAYS :*substrates* (**butane oxygen**)
:*operators* $\mathbf{K}_{\text{free-radical}}$
:*override-environment* **reaction-environment**
:*initiator-p* true)

The difference between these two calls is observed in the **potential reactant queue**: initiation by **sec-butyl-hydroperoxide** results in **butoxy-radical** abstracting **hydrogen** from **butane**; while, forced initiation (i.e. *initiator-p* equals true) cleaves the weakest bond of **butane** forming radicals which abstract **hydrogen** from **butane**. As a consequence, initiation of **butane**, (i.e. formation of the primary chain carrying radical) is the same; although, termination of the initiation products may be different (e.g. ethane versus s-butyl alcohol).

FIND-ALL-PATHWAYS begins by calling composite operators, comprising the domain of free radical operations, on the substrates specified by the keyword argument :*substrates*. These operators loop over the substrate list and evaluate properties specified by $\mathbf{K}_{\text{free-radical}}$. Homolytic dissociation of the substances contained in this listing: **butane**, **oxygen, sec-butyl-hydroperoxide**; is initiated when FEASIBLE-P is evaluated on the sites identified by $\mathbf{K}_{\text{get-sites}}$. These sites are passed to FEASIBLE-P which searches the **reaction-environment** to establish potential radical forming processes. Attributes

describing **reaction-environment** establish thermal cleavage as a potential initiation mechanism; this is shown below:

reaction-environment

temperature: 373 K

wavelength: nil

surface-type: *unbound*

Using this knowledge, $\mathbf{K}_{\text{initiation}}$ constructs a list of potentially-cleavable-bonds by applying IDENTIFY-WEAKEST-BONDS to the bond representation of each substrate contained in the substrate-list. IDENTIFY-WEAKEST-BOND returns a nested list of bonds, arranged in increasing strength.

For each system of reactants, IDENTIFY-WEAKEST-BOND identifies the weakest substrate **bond**, and compares it with the others in the system to ensure minimal global bond strength. Selected **bonds** are then appended to potentially-cleavable-bonds. The weakest system **bond** is identified by applying IDENTIFY-ABSOLUTE-WEAKEST-BOND to potentially-cleavable-bonds. The value of **weakest-bond** is the oxygen-oxygen single bond connecting **oxygen-1** to **oxygen-2** in **sec-butyl-hydroperoxide**. That is:

IDENTIFY-ABSOLUTE-WEAKEST-BOND

applied to potentially-cleavable-bonds ⇒ <**bond-1** "O-O">

Once selected, this **bond** is associated with the global queue **weakest-bond**:

weakest-bond ⇐ <**bond-1** "O-O">

Identification of **weakest-bond** and its bond strength ensures that cleavage will be focused on the weakest bonds of the system as new substrates are added to the reaction system.

After potentially-cleavable-bonds and **weakest-bond** become known to $K_{initiation}$, FEASIBLE-P evaluates the energy in **reaction-environment.** FEASIBLE-P assesses whether **<bond-1** "O-O"> will cleave using SUFFICIENT-THERMAL-ENERGY-P. When FEASIBLE-P evaluates true, $K_{initiation}$ calls K_t on **<bond-1** "O-O">. Application of bond cleavage operations on the weakest bond of **sec-butyl-hydroperoxide** produces the **micro-homolysis-reaction** shown below:

micro-homolysis-reaction

reactants:	**(sec-butyl-hydroperoxide)**
products:	**(sec-butoxy-radical-1 hydroxy-radical-2)**
stoichiometry:	**((sec-butyl-hydroperoxide.** -1)
	(sec-butoxy-radical-1 . +1) **(hydroxy-radical-2** . +1))
bond-cleaved:	**<bond-1** "O-O">
environment:	**reaction-environment**

Successful execution of $K_{initiation}$, forms **micro-homolysis-reaction.** Unique products of **micro-homolysis-reaction** are then collected and appended to the global queue, **potential-reactant-queue**. Substrates contained in **potential-reactant-queue** become potential reactants in future calls on the operations comprising FIND-ALL-PATHWAYS. These operators loop over the substrates of **potential-reactant-queue** and evaluate properties specified by $K_{free\text{-}radical}$. Characterization of **sec-butoxy-radical-1** by **chemical-behavior** restricts the number of composite operations applied to **sec-butoxy-radical-1** due to the reactivity exhibited by **singly-occupied-molecular-orbital.** The enhanced reactivity of this site guides the search of $K_{get\text{-}sites}$.

Ten potential reaction centers are identified by $K_{get\text{-}sites}$ for the attack of **sec-butoxy-radical-1** on **butane.** EQUIVALENT-ATOM-P applied to this list, establishes that all but two reaction centers are unique. These sites are passed to FEASIBLE-P, which calls for an

initial evaluation of the potential reaction centers. Preliminary evaluation of the potential reaction centers suggests the following possible reactions if $\mathbf{K_t}$ was to be executed on them:

1. $C_4H_{10} + C_4H_9O^{\bullet} \rightarrow CH_3{}^{\bullet}CHCH_2CH_3 + C_4H_{10}O$

2. $C_4H_{10} + C_4H_9O^{\bullet} \rightarrow {}^{\bullet}CH_2CH_2CH_2CH_3 + C_4H_{10}O$

Radicals formed from these reactions are the result of hydrogen abstraction. FEASIBLE-P screens this potential reaction list by calling ASSESS-THERMODYNAMIC-FAVORABILITY on the pairs of reaction centers. It passes the most favorable of those reaction centers to $\mathbf{K_t}$. ASSESS-THERMODYNAMIC-FAVORABILITY establishes that abstraction of hydrogen from the secondary carbon of butane is favored over abstraction from the primary carbon; however, since the energy difference resulting from these abstractions is less than 10 kcal/mol both reaction center pairs are passed to $\mathbf{K_t}$.

$\mathbf{K_{propagation}}$ then calls $\mathbf{K_t}$ on the reaction center pairs selected. Application of $\mathbf{K_{ab\text{-}initio}}$ operations to the reaction centers produces a series of **micro-addition-reactions, micro-abstraction-reactions,** and **micro-disproportionation-reactions.** Essential information concerning the **micro-reactions** described above, is shown below:

micro-abstraction-reaction

reactants: **(butane butoxy-radical-1)**

products: **(sec-butyl-radical sec-butanol)**

stoichiometry: **((butane. -1) (butyloxy-radical . -1)**
(sec-butyl-radical . +1) (sec-butanol . +1))

reaction-centers: **(carbon-2 oxygen-1)**

environment: **reaction-environment**

context: **context-1**

The **micro-abstraction-reaction** creates **butyl**, the primary chain carrying radical, from **butane**. Unique products identified by **micro-reaction** are then appended to the global queue **potential-reactant-queue**. Continued application of the composite operations, invoked by FIND-ALL-PATHWAYS, on the substrates contained in **potential-reactant-queue** identifies additional key reactions including those involving oxygen. These are shown below:

micro-addition-reaction

reactants:	**(sec-butyl-radical oxygen)**
products:	**(sec-peroxy-butyl-radical)**
stoichiometry:	**((sec-butyl-radical** . -1) **(oxygen** . -1) **(sec-peroxy-butyl-radical** . +1))
reaction-centers:	**(carbon-2 oxygen-2)**
environment:	**reaction-environment**
context:	**context-1**

Continued application of composite operations, particularly those of $\mathbf{K}_{\text{termination}}$, invoked by FIND-ALL-PATHWAYS, identifies termination of the free radical pathway by disproportionation:

micro-disproportionation-reaction

reactants:	**(sec-peroxy-butyl-radical**)
products:	**(methyl-ethyl-ketone sec-butanol sec-butoxy-radical)**
stoichiometry:	**((sec-peroxy-butyl-radical** . -2) **(oxygen** . +1) **(methyl-ethyl-ketone** . +1) **(sec-butanol** . +1))
reaction-centers:	**(oxygen-2 oxygen-3)**

environment: **reaction-environment**

context: **context-1**

By tracing the history of these reactions using the attribute values of reactants and products, **free-radical-reaction** is able to construct the free radical pathway beginning with initiation and ending with termination. Essential information describing **free-radical-reaction-1** is shown below:

free-radical-reaction-1

reactants: (**butane oxygen sec-butyl-hydroperoxide**)

products: (**methyl-ethyl-ketone sec-butanol sec-butoxy-radical**)

products-formed-in-propagation-cycles: nil

elementary-initiation-reactions: (**initiation-1**)

elementary-propagation-reactions: (**addition-1 abstraction-1**)

elementary-termination-reactions: (**disproportionation-1**)

The pathway originating from these reactions is shown in Figure 6-40. Attributes describing this pathway identify the reactants to be **butane**, **oxygen**, and the initiator **sec-butyl-hydroperoxide**. Non-radical products identified by this pathway are **sec-butanol** and **methyl-ethyl-ketone**. The competing pathway, **pathway-2**, is identified using the methods of **pathway-1** by chaining through the reactants of **pathway-1** and **pathway-2** and finding substrates which are common to both. Composing reactions are identified by linking the reactants to the products for the overall reaction; they are desrcibed by **free-radical-reaction-1.**

Expansion of **pathway-2** shows and alternative route from the reactants, **butane** and **oxygen** to the product, **sec-butanol**. This is shown in Figure 6-41. The competing pathway, **pathway-1**, is established by the occurance of **sec-butylperoxy-radical** in both **pathway-1** and **pathway-2**. Composing reactions of **pathway-2** is **free-radical-**

identifier:	*unbound*
name:	"pathway-1"
reactants:	(#<C_4H_{10}> #<O_2> #<$C_4H_{10}O_2$>)
products:	(#<$CH_3COCH_2CH_3$> #<O_2> #<$C_4H_{10}O$>)
competing-pathways:	(<pathway-2>)
composing-reactions:	(#<free-radical-reaction-1>)
global-rate-expression:	#<composite-rate-exp-1>
global-equilibrium-constant:	*unbound*
context:	#<context-1>

Figure 6-40: Pathway Butane Oxidation: pathway-1 description

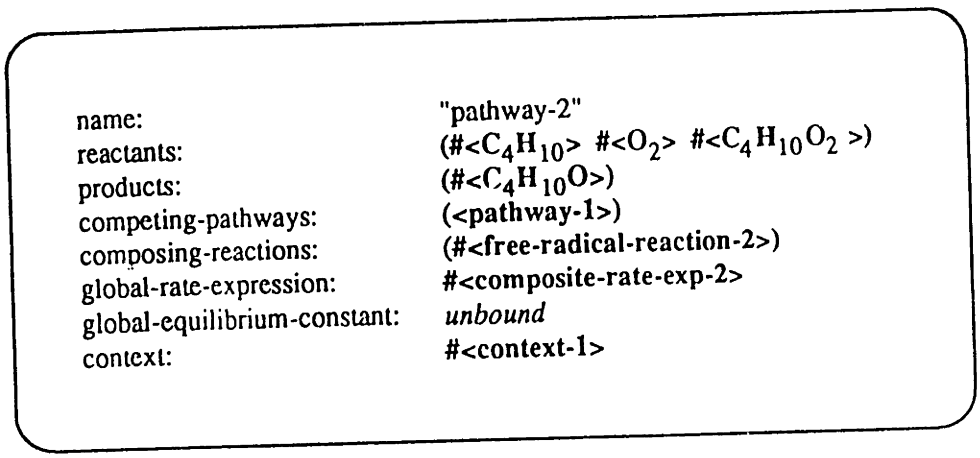


Figure 6-41: Butane Oxidation: pathway-2 description

pathway-2 which is comprised of the same initial reactions of **free-radical-reaction-1**: **initiation-1, abstraction-1,** and **addition-1;** and three additional reactions: **abstraction-2, scission-1,** and **abstraction-3.** The combined result of these two pathways is shown in Figure 6-42 for **context-1.**

Suppose now we wish to create a new context, **context-2,** reflecting a change in **reaction-environment.** CRL manages the change in context by establishing the value changes between **context-1** and **context-2** and by assessing the impact of those changes on FEASIBLE-P. It then reevaluates the conditionals and the functions specifying FEASIBLE-P in the new context to establish alternative behavior. **Context-2** enables **beta-scission** of **sec-butoxy-radical** forming **acyl-radical** and **ethyl-radical** due to an evaluation in reaction temperature. Reevaluation of these reactions in **context-2** leads to the formation of **peracetic-acid;** the state of **potential-reactant-queue** must be maintained to prevent duplication of effort during the reevaluation process.

The reaction pathway enabled by changing **context-1** to **context-2** and the relationships between pathways of **context-1** and pathways of **context-2** are illustrated in Figure 6-43. Figure 6-44 shows the new pathway resulting from **context-2.** Notice that **pathway-1** and **pathway-2** are competing pathways to **pathway-3** and that composing reactions are **free-radical-reaction-1, free-radical-reaction-2,** and **free-radical-reaction-3.** Of those reactions, **free-radical-reaction-1** and **free-radical-reaction-2** are the identical reactions identified in **pathway-1** and **pathway-2,** while **free-radical-reaction-3** represents the additional reactions required to take **sec-butoxy-radical** to **peracetic-acid.** Micro-reactions composing free-radical-reaction-3 include: **scission-2, addition-2, disproportionation-2, abstraction-4, addition-3,** and **abstraction-5.** These reactions are all related to **context-2** through the value of the attribute context describing pathway.

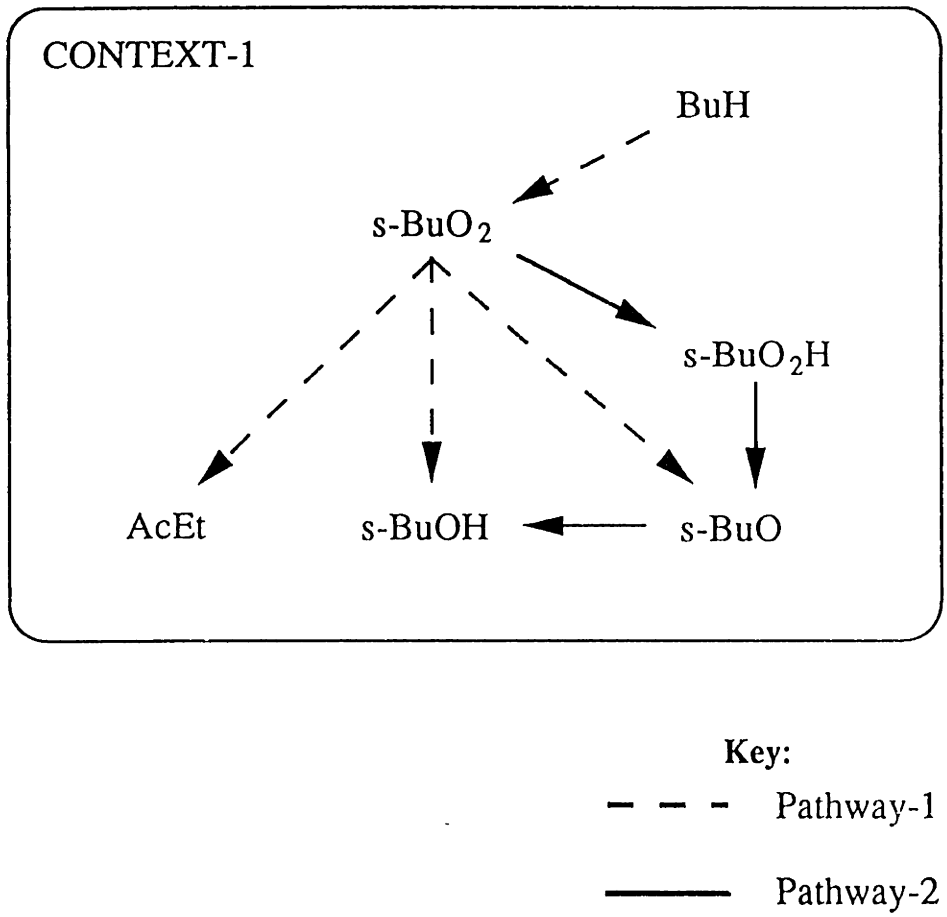


Figure 6-42: Oxidation pathways of butane

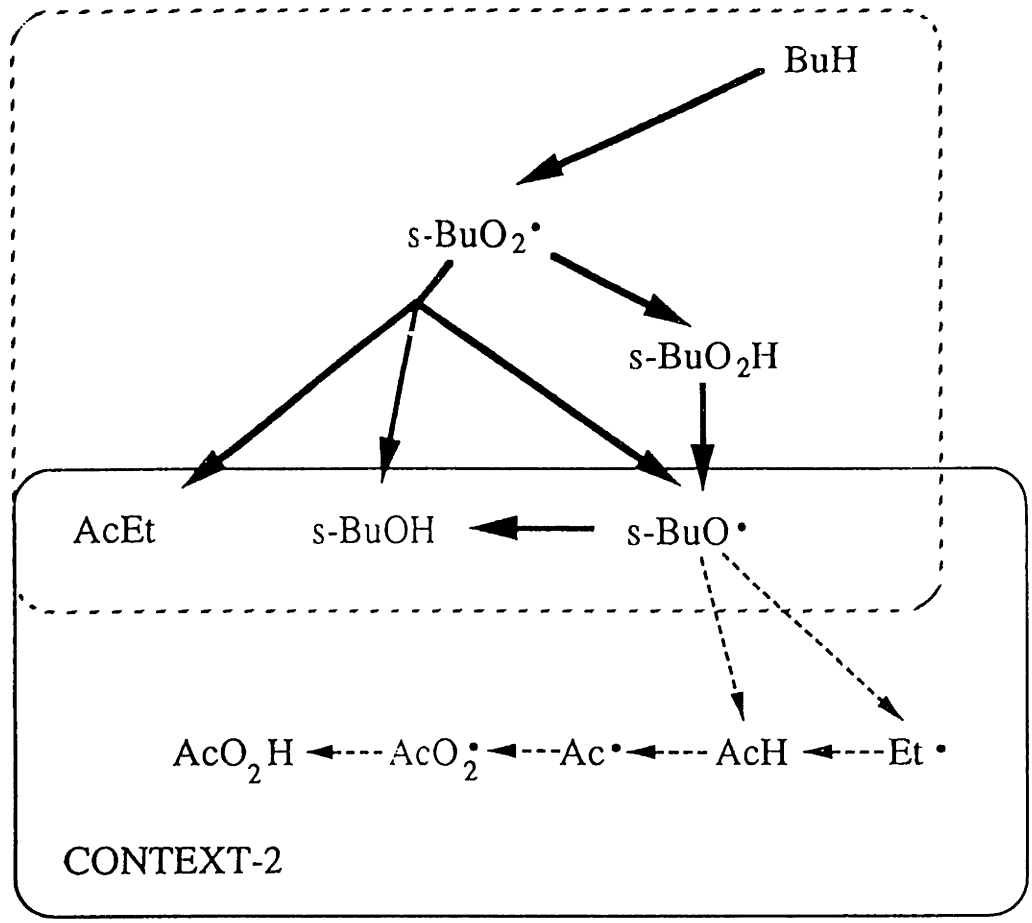


Figure 6-43: Oxidation pathways of butane: Effect of changing context

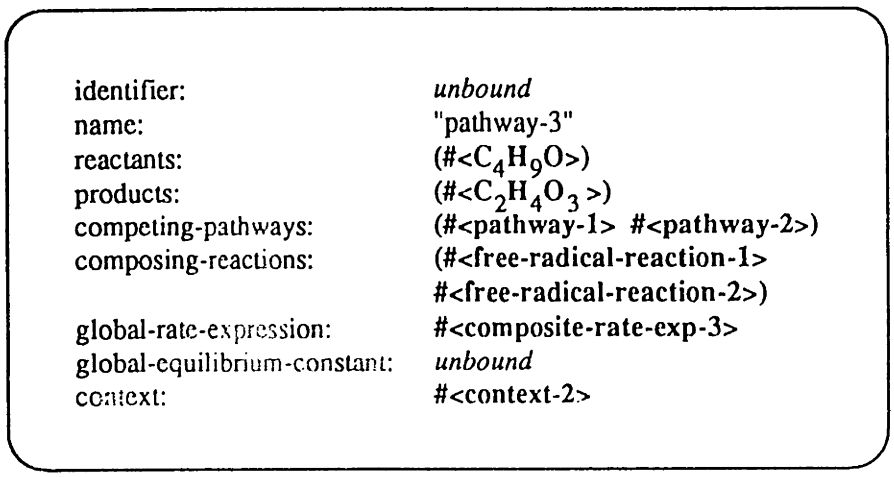


Figure 6-44: Butane Oxidation: context-2 pathways

6.7 Conclusions

A language for modeling chemical behavior and reasoning about chemical systems, called CRL, has been presented. Realizing the limitations of the previous procedural attempts, it is based on an object-oriented, declarative approach. The language has been engineered to: (i) discover new synthetic pathways; (ii) decompose smoothly from theoretically bounded reactions to known reactions, (iii) facilitate rapid and easy extension to new reaction classes; (iv) learn to enhance search efficiency; (v) capture and utilize qualitative, semi-quantitative relationships (ordinal, order-of-magnitude), or boolean relationships; (vi) offer explicit documentation of all the hypotheses, assumptions and simplifications that give rise to a particular model, pathway, or reaction. CRL can be viewed as a very high level special purpose language, that moves the user several levels away from the inherent programming language (e.g. Lisp in this case).

CRL's characteristics have been extensively discussed. After introducing the language's components, the construction of models has been formally addressed, with specific references to its grammar and semantics. CRL has an English like syntax: very little training is required to write the specifications of the model class. This feature makes it possible for all potential users of the language to participate actively in the specification of new modeling classes or in the modification of existing ones.

CRL complies with all the requirement that were proposed for its design. Its strengths are its modularity and its inherent capability of controlling complexity by breaking apart complex chemical systems piece by piece, into smaller, less complex pieces. The specification of a model class involves subsequent specification and characterization of all its components. This idea is recursively applied throughout the definition process. Thus, it is similar to describing a synthetic plan or a complex chemical system in a natural language. Another important feature of this language is its extensibility; the ability to characterize a chemical system can be extended by incorporating a richer

terminal vocabulary. Its modularity makes possible the incorporation of new blocks in the definition of a class; blocks that may describe other aspects that are presently not considered. CRL supports the creation of new data types and modeling elements.

6.8 Bibliography

1. Simon, H. A., "The Sciences of the Artificial," M.I.T. Press, Cambridge, MA (1969)

2. E. J. Corey, *Pure Appl. Chem.*, 14, 19-37 (1967).

3. "Computer-Assisted Organic Synthesis," (W.T. Wipke and W.J. Howe, eds) *ACS Symposium Series*, 61, Washingtom DC (1977).

4. Joback, K. G.," Designing Molecules Possessing Desired Physical Property Values," M.I.T. Doctoral Thesis (1989)

5. Wolf, W. H., T. J. Kowalsky, and M. C. McFarland, "Knowledge Engineering Issues in VLSI Synthesis," Proceediings AAAI 86, 8660887 (1986)

6. Stephanopoulos, G. and Th. Kriticos, "An Artificial Intelligence Perspective in the Design of Chemical Processes," AIChE Meeting, Miami Beach, FL (1986)

7. Lakshmanan, R. and G. Stephanopoulos, "Synthesis of Operating Procedures for Complete Chemical Plants, Part I: Hierarchical, Structured Modelling for Nonlinear Planning," Comp. Chem. Engng 9/10, 985-1001 (1988)

8. Johnston, J. and G. Stephanopoulos, "Synthesis of Plant-Wide Control Configurations with a Computer-Based Design Expertt: Parts I, II, III, IV, Working Papers, MIT-LISPE, Dept. of Chemical Engineering (1989)

9. Henning, H., Leone, H., and Stephanopoulos, G, "MODEL.LA: A Modeling Language for Process Engineering, Part I: The Formal Framework," Comp. Chem Engng., **14**(8) p. 813 (1990)

10. Douglas, J., "Conceptual Design of Chemical Processes," McGraw-Hill (1988)

11. Sussman, G. and G. L. Steele, "CONSTRAINTS - A Language for Expressing Almost Hierarchical Descriptions," Artificial Intelligence, 14, 1-39 (1980)

12. Weiss, S., C. A. Kulikowski, S. Amarel, and A. Safir, "A Model-Based Method for Computer-Aided Medical Decision-Making," Artificial Intelligence, 11, 145-172 (1978)

13. Ireland, R. E., "Organic Synthesis," Prentice-Hall, NJ, (1969)

14. Cormen, T. H., C. E. Leiserson, R. L. Rivest, "Introduction to Algorithms," M.I.T. Press, Cambridge, MA (1990)

15. Steele Jr., G. L., "Common Lisp - The Language," Second Edition, Digital Press, USA, (1990)

16. Dugundji, James, and Ivar Ugi, "An Algebraic Model of Constitutional Chemistry as a Basis for Chemical Computer Programs," *Top. in Curr. Chem.* 39(19) (1973)

Appendix A

Axiom -- Generalized Transitivity

In order to introduce the generalized transitivity axiom, let us first define the ordering relation "^{3}" for the semantic relations defined in CRL. The relation x^3y, that is read as "x is superior or equal to y", is defined by listing all pairs for which it holds. In CRL the semantic relations have been defined to obey the following ordering relationships:

IS-DISAGGREGATED-IN	≥	IS-COMPOSED-OF
IS-COMPOSED-OF	≥	IS-CONNECTED-BY
IS-CONNECTED-BY	>	IS-DESCRIBED-BY
IS-ABSTRACTING	>	IS-PART-OF
IS-PART-OF	≥	IS-ATTACHED-TO
IS-ATTACHED-TO	≥	IS-DESCRIBING
SEMANTIC-RELATION-j	≥	SEMANTIC-RELATION-j

This ordering relation can be characterized as reflexive, anti-symmetric and transitive. Then, we can deduce that CRL's semantic relations were created to obey the generalized transitivity axiom, that can be stated as follows:

If (O1 "*semantic-relation-1*" O2) and (O2 "*semantic-relation-2*" O3)

then

(O1 "*least element of A*" O3)

where A = {*semantic-relation-1*, *semantic-relation-2*}

A is partially ordered by "≥"

and "*least element of A*" = a/a ∈ A ∧ x ≥ a for all x ∈ A

In order to illustrate the axiom, let us assume that "*semantic-relation-1*" is "***is-composed-of***", "*semantic-relation-2*" is "***is-described-by***", O1 is a reaction, R-1, O2 are

products, Products-R-1, and O3 is the variable TEMPERATURE-PRODUCTS-R-1, then we can write:

If (R-1 ***is-composed-of*** Products-R-1)

and (Products-R-1 ***is-described-by*** TEMPERATURE-PRODUCTS-R-1)

then (R-1 ***is-described-by*** TEMPERATURE-PRODUCTS-R-1)

It is self evident that this axiom can be recursively applied several times. For instance, if we know that the pathway, P-1, ***is-composed-of*** R-1, and we have asserted that R-1 ***is-described-by*** TEMPERATURE-PRODUCTS-R-1, we can easily conclude that P-1 ***is-described-by*** TEMPERATURE-PRODUCTS-R-1.

Let us now consider the example considered in the oxidation of butane. When going from the level of butane pathway to the individual reaction sets the following disaggregation is performed:

Butane-Pathway ***is-disaggregated-in*** (SET.OF (Initiation, Propagation, Termination))

At the reaction set level it is known that: (i) Propagation ***is-connected-by*** Propagation-Radicals to Termination, and (ii) Termination ***is-described-by*** the TEMPERATURE-PRODUCTS-OF-TERMINATION variable. Then, by the generalized transitivity axiom it is possible to assert that the Butane-pathway ***is-described-by*** the variable Products-of-Termination.

Appendix B

OBTAIN-VALUE (VARIABLE $CONTEXT_i$ $CONTEXT_j$)

IF (= $CONTEXT_i$ $CONTEXT_j$)

THEN (RETURN VARIABLE-VALUE IN $CONTEXT_i$)

ELSE IF PATH = (FIND-PATH $CONTEXT_i$ $CONTEXT_j$)

THEN $CONTEXT_{j-1}$ = (NEAREST-NEIGHBOR-IN-PATH $CONTEXT_i$ $CONTEXT_j$)

IF VARIABLE $\exists$ IN $CONTEXT_{j-1}$

THEN (OBTAIN-VALUE VARIABLE $CONTEXT_i$ $CONTEXT_{j-1}$)

ELSE IF [OR (VARIABLE *is-disaggregated-in* {?X} IN $CONTEXT_{j-1}$)]

([VARIABLE *is-abstracted-by* ?X IN $CONTEXT_{j-1}$)]

THEN (OBTAIN-VALUE VARIABLE $CONTEXT_i$ $CONTEXT_{j-1}$)

ELSE IF $\exists$ $MF_{CONTEXTj-1}$/VARIABLE-VALUE = $MF_{CONTEXTj-1}$(Y-VALUE)

$\wedge$Y = {Y/Y *is-abstracted-by* VARIABLE IN $CONTEXT_{j-1}$}

THEN EVAL ($MF_{CONTEXTj-1}$(Y-VALUE), Y-VALUE = {Y-VALUE/Y-VALUE = (OBTAIN-VALUE Y $CONTEXT_i$ $CONTEXT_j$) Y$\in$Y})

ELSE

(RETURN "There is no feasible path to transfer information")

END IF

END IF

END OBTAIN-VALUE

Chapter 7
Conclusions and Recommendations

7.1 Conclusions

An intensive, process based methodology has been developed to solve the problem of identifying inherent design weaknesses. The methodology is complete within the scope of modeling efforts. The approach begins with the inductive determination of hazardous chemical or physical reactions, or both, and proceeds deductively to the network of processing steps to identify all the root causes initiating a chain of events leading to hazards.

The approach is evolutionary and is based on chemistry and engineering science rather than collective team judgment. As such, quantification can be used at any stage of the design process to discriminate among design improvements. More importantly, since the use of available knowledge is maximized at each phase in the design sequence, this affords the earliest possible identification of hazards.

These contributions have led to the development of a computer language for reasoning about chemical structures. This language supports the automation of hazard identification. Methods developed using this language afford automatic generation of reactant products from a given set of substrates. The language has been designed to facilitate rapid development and is easily extendable to new reaction classes.

Specific contributions include:

1. Establishment of a formal strategy for the integration of safety into a design technology.
2. Establishment of a means for discriminating among design alternatives with respect to disturbance mitigation.
3. Development of a basis for the optimization of a design technology with respect to disturbance mitigation.

4. Establishment of a means for automatically constructing the network of pathways leading to a hazardous state.

5. Development of a means for creating qualitative fault trees from the information associated with the hazardous state and the pathway leading to it.

7.2 Recommendations

It is recommended that future work be directed at the following specific requirements relating to the identification of hazards:

1. Integration of quantitative assessment techniques into the methodology to limit qualitative pathways.

2. Investigation of the effects of input set assignment on pathway topography.

3. Determination of the effect of equipment placement and its selection among design alternatives and their relationship to the inherent safety of the design technology.

4. Refine the development of qualitative fault tree construction and develop the mechanism for transforming qualitative fault trees into quantitative fault trees.

5. Develop the means for associating external knowledge bases with the methodology to enhance the conversion of qualitative fault trees (e.g. knowledge bases which can infer that an operator failed to respond).

6. Extend CRL to include new reaction classes.