

CONF-840113--1

CONF-840113--1

AUTOMATED REASONING APPLICATIONS TO DESIGN VALIDATION AND  
SNEAK FUNCTION ANALYSIS

DE84 003994

by

R. C. Stratton

EBR-II Project  
Argonne National Laboratory  
P.O. Box 2528  
Idaho Falls, Idaho 83401

The submitted manuscript has been authored by a contractor of the U. S. Government under contract No. W-31-109-ENG-38. Accordingly, the U. S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or allow others to do so, for U. S. Government purposes.

Submitted for Presentation  
at the  
1st Symposium on Space Nuclear Power Systems  
January 10-13, 1984  
Albuquerque, New Mexico

**NOTICE**  
**PORTIONS OF THIS REPORT ARE ILLEGIBLE.**  
It has been reproduced from the best available copy to permit the broadest possible availability.

\* Work Supported by the U.S. Department of Energy  
under Contract W-31-109-38.

**DISCLAIMER**

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

**MASTER**

DISTRIBUTION OF THIS DOCUMENT IS UNLIMITED

## INTRODUCTION

Argonne National Laboratory (ANL) is actively involved in the LMFBR Man-Machine Integration (MMI) Safety Program. The objective of this program is to enhance the operational safety and reliability of fast-breeder reactors by optimum integration of men and machines through the application of human factors principles and control engineering to the design, operation, and the control environment, Vaurio, et al (1982). Similar objectives pertain to other nuclear systems including space nuclear plants. Space nuclear power systems must be designed, developed, and controlled with a high degree of integrity if they are to fulfill goals of long-term reliability. Technology being developed at Argonne for the LMFBR-MMI program is directly applicable to the space reactor program since many of the concerns associated with high reliability and safety of control systems over long operating lifetimes are common to both. One aspect of the MMI work at ANL, and its implications, is reported in this paper.

Validating that the system design function incarnation (the physical product of design) matches the functional requirements and identifying sneak functions (unplanned element functions) are necessary for ensuring that a system meets reliability criteria, Wojcik (1983) and Ehrlich (1983). Classically, validation and sneak function analysis are performed by the design engineer. The effectiveness of this validation and analysis is directly proportional to the engineer's training, experience, and thoroughness. Generally, the better trained and the more experienced the engineer, the more functional the design and hence, the better the transition to the physical incarnation with reduced or eliminated sneak functions.

ANL is developing methods to apply automated reasoning and computerization in the validation and sneak function analysis process. If the proper relationships and definitions of design functions and components are provided, Stratton et al (1983) and Fabriel (1983), then validation of system incarnation and sneak function analysis can be achieved via automated reasoners (AR) such as Logic Machine Architecture, Lusk et al (1982), and Prolog, Clocksin et al (1981) and Kowalski (1982).

This project provides the element definitions and relations necessary for an automated reasoner (AR) to reason about design validation and sneak function analysis. This project also provides a demonstration of this AR application on an Experimental Breeder Reactor-II (EBR-II) system, the Argonne Cooling System (ACS). The initial demonstration will be limited to one of the subfunctions provided by the ACS, the reactor subassembly cooling function. This will be discussed in greater detail in later sections dealing with performing validation and sneak function analysis.

## RELATIONSHIPS AND DEFINITIONS

Element relationships and definitions are required to define the system model and rules for the automated reasoner. Elements are the physical entities that comprise the total process and are defined as

components, subsystems, systems, and the process. The relationships and definitions are manifested as element functional definitions, state relationship to functions, function relationship to direction, element linkage, and functional hierarchical configuration. "State" defines the operability and readiness of an element to perform a function. "Operability" classifies the capability of an element to be operated, and "readiness" describes the element's closeness to the next desired state, Seeman et al (1982) and Colley (1982). These relationships and definitions are further defined below.

Justification for elements in a system is that elements satisfy functions. Each element, therefore, represents single or multiple functions i.e. turbines and heat exchanger, respectively (Fig. 1). These functions exist in a hierarchical network of functions that satisfy a design function. The elements can be characterized as providing functions at required design capacity or limits. This paper deals with the functional aspect of the element. Future research will examine and incorporate the design capacity issues. The functional aspect of elements can best be understood by examples such as pumps, valves, and filters. As a component element, the function of a pump is to develop a driving force (a pressure differential), the function of a filter is to purify a media, and the function of a valve is to regulate flow. As an example of a subsystem element, the combined functions of a pump, filter, and valve can be to prevent damage to a related subsystem or component via media purification.

Element state has a direct relationship to element function. Therefore the element function is further defined or bounded by the state of the element. An element with multiple functions is capable of providing a defined function only when it resides in a specific state, i.e., on, off, variable, etc. and the element may provide certain functions when in one state while providing other functions in yet another state. As an example, a valve can provide the functions of isolation, path connection, and regulation. The isolation function exists only when the valve is in the "off" state (shut) whereas the path and regulation functions exist when the valve is in the "on" or "variable" state.

In addition to element functions being bounded by the state, the functions are also bounded by media flow direction. The direction of flow through an element can alter the element's functional characteristics. The alteration can either be neutral, negative or positive relative to the element's function. Using a valve and a filter as an example, the directional dependence of function can be further explained. A check valve will provide flow in one direction and isolation in the opposite direction, whereas a gate valve in the "open" state will provide "path" in both directions. However, a radioactive particulate filter can provide filtration in one direction and provide contamination in the other direction. Obviously, the filtration function is a desired (positive) function whereas the contamination is an undesirable (negative) function.

Element linkage characterizes the physical connection between elements. Linkage defines the serial and parallel functions of the process as well as the function integration. Linkage provides the mechanism that allows

FUNCTIONAL HIERARCHY

ELEMENT TYPE

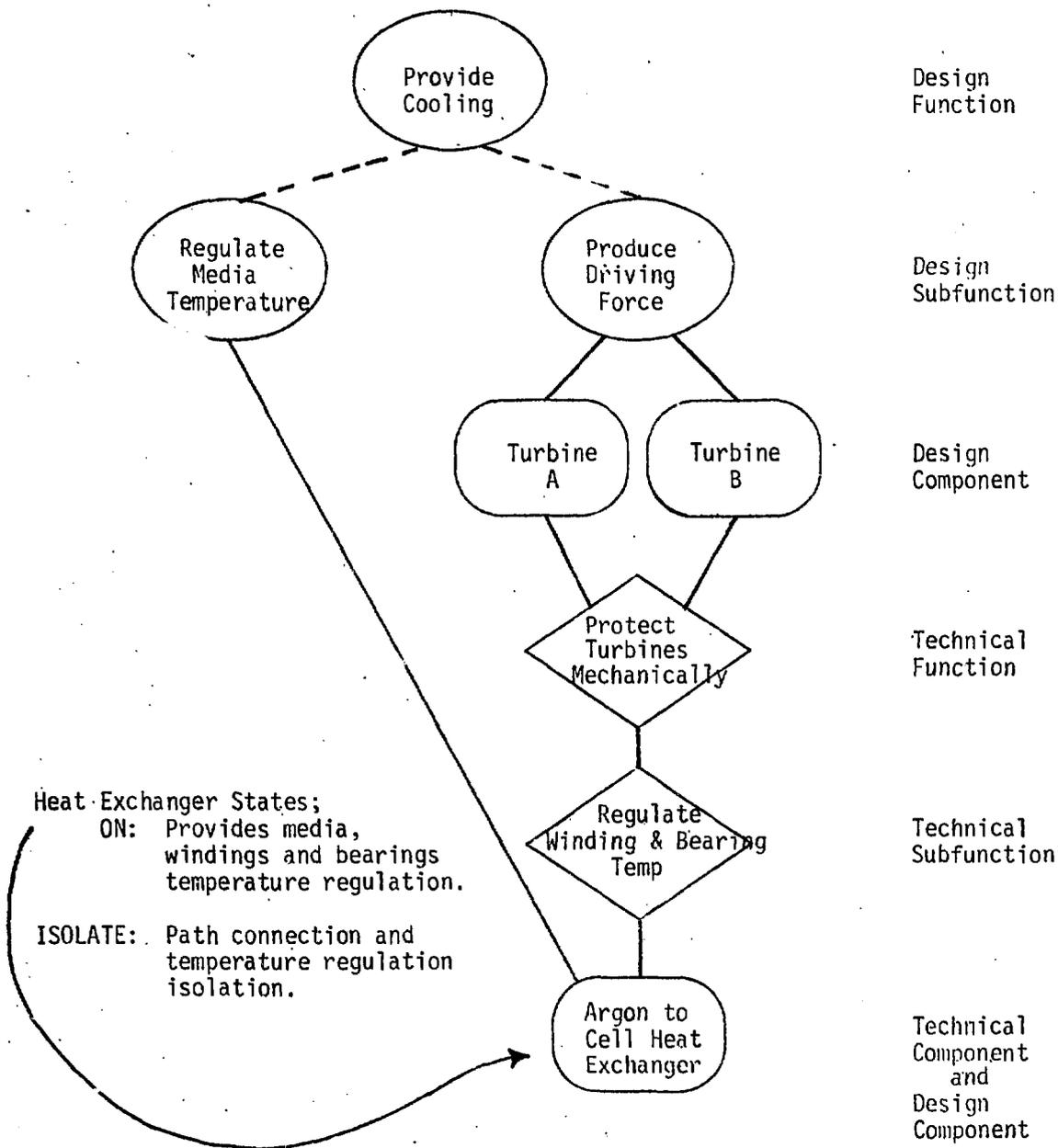


Figure 1. HIERARCHY, ELEMENTS AND STATE RELATIONSHIPS

function evaluation of the integrated elements for comparison with the design function requirements.

The functional hierarchical configuration describes the functional relationships necessary to provide for the overall (top level) design function. The top level design function is the highest node in the functional hierarchy. Each successively lower level of the hierarchy defines sibling functions required to build the parent or top level function. Functions are classified as either design functions or technical functions. Design functions are independent of physical incarnation and express the physics and engineering functions necessary to satisfy the design function. Technical functions are directly dependent on physical incarnation and exist because of the inadequacies of technology. See Fig 2.

### Validation and Sneak Function Analysis

Validation is the act, process, or instance of assuring the compliance of an object to a standard. "Validation," as used herein, is used to mean the process that consists of verifying that the design incarnation functionally complies with the design specification properties. Therefore, the product of this validation ensures that, at a minimum, the incarnation provides for the functions designated in the design specification.

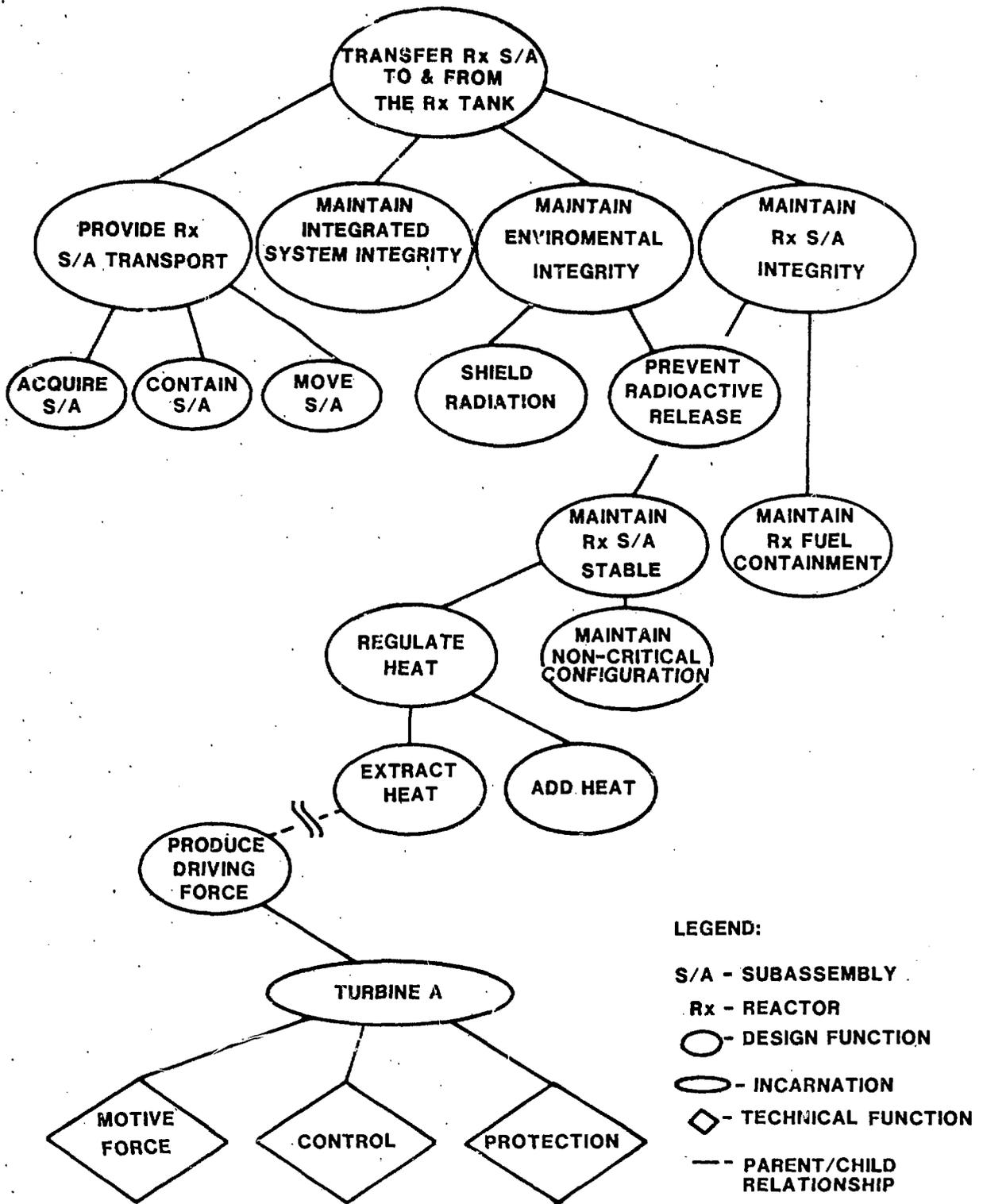
Validation is performed by a direct functional comparison of the design incarnation to the design specification. The entities; specification, and incarnation, are expressed as functional hierarchies by examining, evaluating and transposing their parts into functional constituents. The hierarchies are then compared to determine agreement and disagreement for validation.

Sneak function analysis is the reasoning about an incarnation to determine unplanned element functions as a result of the specific incarnation. The analysis is performed by analyzing the functional hierarchy, as defined by the incarnation, for paths that provide functions not specified by the design function or not specifically specified by the design engineer. The sneak functions are then analyzed to determine the benefit and/or detriment to the system as a whole.

### Automated Reasoner Algorithms

AR algorithms in this project provide for the generation and association of element parameters (state, function, and type) and functional hierarchies. AR algorithms also provide for validation and sneak function analysis. These algorithms are divided in the following categories; knowledge base, path, path function, validation, and sneak function analysis.

The knowledge base algorithms generate and maintain the information that defines the physical incarnation. This information is represented as graph arcs where each arc encapsulates the data that defines the arc



**FIGURE 2: EXAMPLE OF A FUNCTIONAL HIERARCHY (FUNCTIONS NOT COMPLETE) SHOWING PARENT/CHILD RELATIONSHIP, DESIGN FUNCTIONS, AND TECHNICAL FUNCTIONS**

and arc function. The factor that defines the characteristics of the arc is the component element that resides within the arc. The knowledge base algorithms use the information concerning the component and component type to determine the arc's functions as dependent on state and direction.

The design incarnation appears to the engineer as a piping and instrument diagram (P&ID) (Fig. 3). For the purpose of validation and sneak function analysis, the P&ID is transposed into a graph composed of vertices and arcs. Each arc defines a design incarnation (as discussed above) and maintains the relationships of arcs through linkage (and therefore the relationships of the elements). The functions provided by the incarnation are determined via an exhaustive search and analysis of all possible paths in the graph. These paths are determined by the path algorithm. Given a start vertex and a goal vertex, the path algorithm marches from vertex to vertex determining connection based on arc state and direction. The process is continued recursively until all paths have been determined.

The function of each path is determined by the path function algorithms. The path function is defined by the sum and relation of the arc functions that exist in the path. The path function algorithms analyze the summed arc functions and derives a path function based on physics and engineering principles. The path function is then compared to the design function for validation and/or analyzed as a sneak function.

Path functions, which represent the design incarnation, are compared to the design function for validation by the validation algorithm. The validation algorithm employs direct comparison and cononicalization principles (order dependency) to verify agreement of functions and function interrelations between the design function and the functional incarnation. Here the validation algorithm pays specific consideration to the type of design function versus technical functions employed by the incarnation. This consideration is satisfied via the hierarchical structure of the functional incarnation. A path function satisfies validation only if it equals the design function and is specifically designated by the design engineers for that design function purpose. Otherwise, the path function becomes a sneak function.

Determination of sneak functions involves both the validation algorithms and the sneak function algorithms. If a functional incarnation does not validate against the design function, it is then labeled as a sneak function. The sneak function algorithms analyzes the path function to determine the nature of the sneak function and its impact on the design function. The result of the analysis is output to the engineer for concurrence and possibly further analysis or design change.

### ASC System Description

The Argonne Cooling System (ACS) performs a subfunction for the Fuel Handling System (FHS). The FHS functions to transfer nonirradiated fuel into the reactor core and irradiated fuel from the core to the fuel



subassembly shipping cask. The ACS principally functions to provide thermodynamic integrity of the subassembly when it is external to the reactor pool. The ACS circulates argon gas through fueled subassemblies for preheating (during transfers into the primary tank) or cooling (during transfers out of the primary tank).

Functional considerations and constraints that must be understood and accounted for in the design of the ACS are as follows. Subassemblies are either irradiated or nonirradiated. Irradiated subassemblies are radioactive, generate decay heat, and contain radioactive sodium residue. Nonirradiated subassemblies require preheating prior to immersion into the reactor pool sodium environment. The refueling environment contains sodium which is radioactive, highly reactive, and will foul and plateout on system components. All components that come into contact with sodium must subsequently be cleaned of sodium residue. The purity of the primary pool argon cover gas environment must be maintained. Argon must not be released to the atmosphere external to the FHS. The integrity of all interfacing processes must be maintained.

The above functions and constraints must be included in the design specification. The design specification will express the essential functions and incarnations required to provide the process. The functions will be of two classes, design functions and technical functions. The manifestations of the design specification are the physical components and structure of the ACS.

The ACS has three principle flow paths: (1) Fuel Transfer Port (FTP); (2) Fuel Unloading Machine (FUM); and (3) Interbuilding Coffin (IBC). Refer to Fig. 3. Each flow path is designed to provide argon gas flow to a subassembly for heating or cooling as well as providing the necessary ACS support functions. The ACS consists of the following: control console, turbines, heat exchanger, heaters, vapor traps, molecular sieves, gas purifier bed, oxygen monitor, moisture monitor, and pressure control.

The ACS is controlled via the ACS control console which has visual displays of the ACS flow paths, instruments and controls provides the operator with the information necessary for proper remote operation of the system.

The ACS delta pressure (flow) is provided for via the ac or dc turbine. The ac turbine provides flow during normal operation and the dc turbine is used for backup during ac turbine or ac power off-normal events. On loss of normal power, the dc turbine starts automatically and is supplied by the ACS batteries until the 400 kW diesel generator assumes the emergency loads. At this time, the dc turbine automatically switches to the rectified ac power supply. Vibration monitors are mounted on the ac and dc turbine flanges. These monitors will provide warning of abnormal vibration to permit turbine switchover or ACS shutdown prior to actual turbine failure. A multiple tube argon-to-air heat exchanger located upstream of the turbines cools the argon to below 100°F. The inlet gas

temperature rating to both turbines is 150°F (maximum) to prevent overheating of the turbine windings and bearings.

The heat exchanger rejects ACS heat to the reactor containment atmosphere. The heat exchanger is a natural convection, air-cooled, horizontal multi-tube type with a design capacity for cooling 70 cfm argon from 700°F to 100°F. In actual operation, the outlet gas temperature is reduced to nearly ambient room temperature.

Four heaters are provided in the ACS for argon and component preheat: gas-purifier bed heaters, FUM inlet-line heaters, FTP pipeline heaters, and subassembly heaters. The gas-purifier bed is equipped with nine resistance heaters rated at 1000 watts each. These heaters are left "Off" to prevent high IBC outlet temperature when the ACS flow is routed through an IBC. The FUM-inlet line heater is an immersion-type heater rated at 9 kW. This heater warms argon entering the FUM and ensures that the subassembly bond sodium is melted from the top down. The FTP path of the ACS has two separate heaters. The pipe line heater consists of two resistance wire-wrapped heaters wrapped around the pipe leading to the FTP. This heater is used to preheat the argon piping. The subassembly heater, an immersion-type heater rated at 12 kW, is located in the piping leading directly to the FTP. Both these heaters heat argon gas to approximately 650°F before it enters the FTP for subassembly blowing and heating sequences. These heaters serve to heat the subassembly to a minimum of 450°F before it is lowered into the primary tank, to limit thermal shock to the subassembly.

The ACS utilizes two vapor traps and two molecular sieves that remove radioactive sodium aerosol from the argon gas preventing accumulations of sodium or its oxide. As the vapor trap in service becomes plugged, the standby unit can be put in service and the plugged one removed for cleaning. However, both molecular sieves are required on-line simultaneously. If either of the molecular sieves becomes plugged, the ACS must be shut down to replace the plugged component. The FUM exhaust line also contains a vapor trap in series with a molecular sieve. These components are similar in construction but physically smaller than the ones described above. The gas purifier bed is filled with stainless steel mesh. The mesh removes particles that would otherwise be circulated through the system.

The oxygen content of the argon must be monitored continuously during fuel-handling operations. Oxygen inleakage will cause sodium oxide formation in the subassembly being cooled. This can block cooling circulation and cause fuel pin melting. Also, oxide formation on the FUM gripper will hinder operation of the jaws and the sensor rod. Two oxygen analyzers are provided to monitor the oxygen content of either the IBC or the ACS. A moisture monitor is provided to sample ACS flow for moisture content. Moisture in the system has the same effect as oxygen. The primary sources of moisture are the IBC's when they are returned from the Hot Fuels Examination Facility.

To ensure that oxygen inleakage in the ACS is held to a minimum, the system pressure is maintained positive when flow is through the FUM only and is therefore assumed positive for all other flow paths. This is accomplished by maintaining the turbine inlet pressure slightly positive which results in a turbine discharge pressure of approximately 3.5 to 4.5 psig. A turbine delta pressure signal is used to control the turbine inlet pressure. This is done by adjusting the throttle valves, which lead to a pressure balance chamber between the turbine inlet and discharge lines. An argon makeup arrangement automatically feeds argon into the ACS when pressure in the chamber decreases to less than two inches of water. It stops supplying argon to the system when the chamber pressure reaches 6 inches of water. The argon makeup and vent subsystem is also used to prevent system damage during ACS heatup and cooldown.

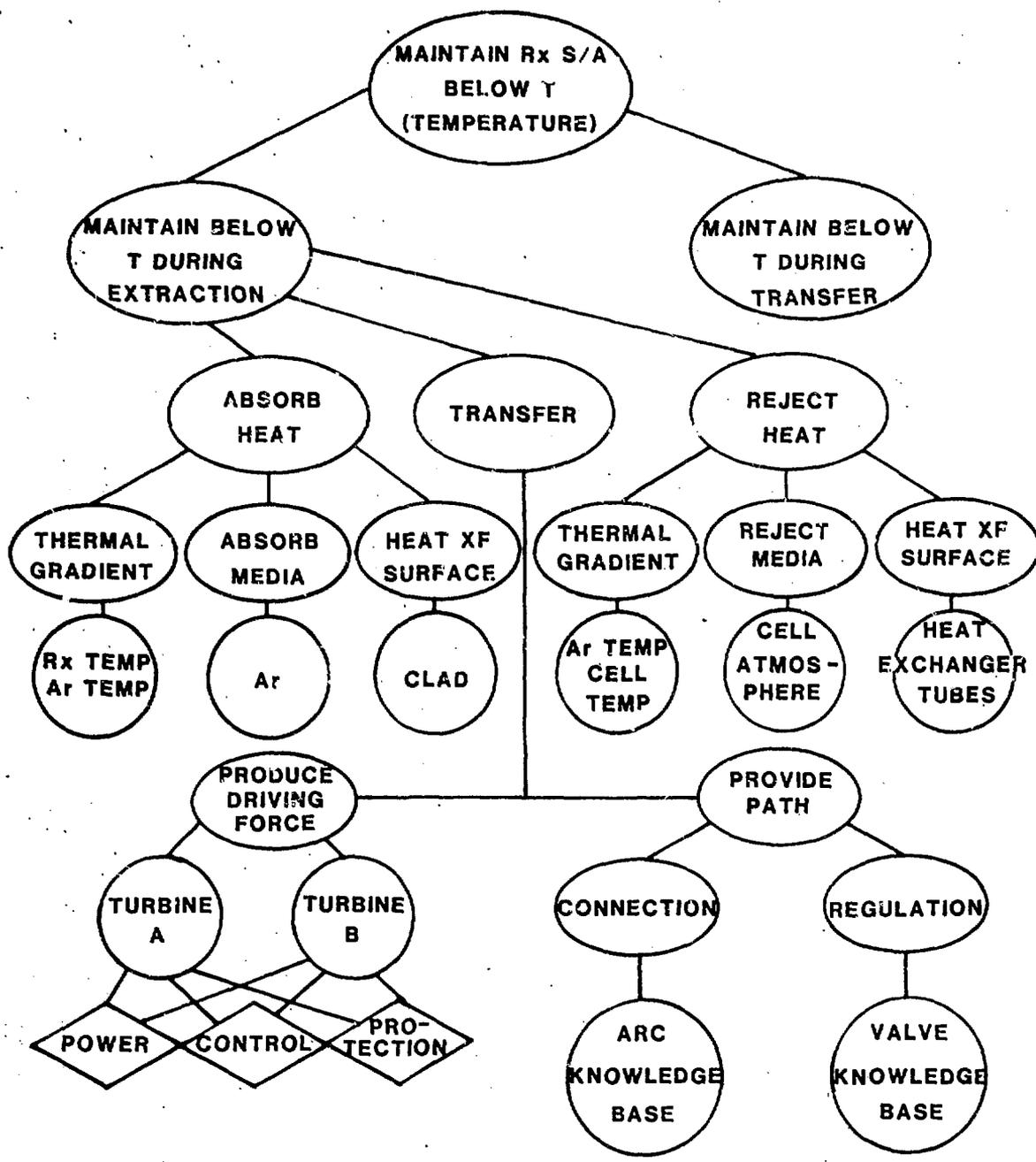
### Performing Validation and Sneak Function Analysis

The procedure for performing validation and sneak function analysis for an incarnated design function is as follows. Specify and represent the design function in a hierarchy based on physics, engineering principles, objectives, and constraints. Represent the incarnation of the design function as an extension of the design functional hierarchy (Fig. 4). Define each element of the incarnation with respect to its functional attributes and interelement relationships (Fig. 1). Then build an incarnated functional hierarchy using the graph representation of the incarnation and the AR application algorithms. And finally, compare and analyze the incarnated hierarchy with the design hierarchy to determine validation properties and sneak functions.

Processes that perform functions are the result of a design function based on physics and engineering principles and bounded by process objectives and constraints. A generic cooling function will require a heat transfer surface, a temperature gradient between the heat-rejecting media and heat-absorbing media, a heat-absorbing media, and a function that will replace the heated absorption media with a cool (relative to the temperature gradient requirements) absorption media. Objectives and constraints in the form of cost, schedule, environment, reliability, etc., will further add to the functional requirements of the process.

One design function of the ACS is to maintain the temperature of a reactor subassembly below a specified temperature when the subassembly is not immersed in sodium. The physics and engineering functions must be as described above. In addition, objectives and constraints are placed on the ACS in that the cooling media is required to be a gas (because of economics and corrosion consideration), the cooling system is to be a recirculation type with redundant delta pressure producers and maintain environmental integrity. Part of the design functional hierarchy for the above ACS function is as shown in Fig. 4.

The engineering incarnation of the design function is represented in the form of component elements. Figure 4 identifies the physical incarnation of the cooling design function for the ACS as an extension of



**LEGEND:**  
 ○ - DESIGN FUNCTION  
 ○ - INCARNATION  
 ◇ - TECHNICAL FUNCTION  
 Rx - REACTOR  
 S/A - SUBASSEMBLY

FIGURE 4: FUNCTION HIERARCHY FOR THE ACS COOLING FUNCTION

the design function hierarchy. It now remains to build an incarnated functional hierarchy as defined by the ACS piping and instrument diagram, Figure 3. Then analysis can be performed regarding validation and sneak functions.

For analysis, the ACS is transposed into arcs as shown in Fig. 5. Each arc is analyzed for its functional attributes as defined by the component element that resides in the arc. In this manner, the graphic representation of the incarnated function is built and stored in the knowledge base by the following prolog statements.

```
arc (vertex 1, vertex 2, component, type)
isarc (vertex 1, vertex 2, component, type, function, state)
```

In order to analyze the incarnated functions as represented by the graph, path and path function algorithms are executed. Paths are determined in the normal graphic sense and stated as follows:

```
path (X,Y,Z) where Z = [v1, v2, v3...vn].
```

The prolog statement "path" asserts the requirement to determine a path for vertex X to vertex Y as defined by the path list Z and the Z vector is a listing of the vertices (V<sub>n</sub>) that define the path in path order.

Each arc, as described by the vertices in the path, is analyzed to determine its function via isarc. The sum of these arc functions then define the path function. The path function is further refined to designate those functions that are the principle objects of the element (design functions) from the function required because of technology (technical functions). An example of a technical function in the ACS is the heat sink prior to the turbines in order to prevent the turbine motor windings and bearings from overheating. Using the refined path function, validation and sneak function analysis can now be performed and the technical functions are noted as per their functional requirement.

In validation, the path function is configured into allowable hierarchical representations. These representations are then compared against two objects, the design functional hierarchy, and the design engineer's intent. If the comparison is acceptable with respect to both objects, then validation can be assigned as performed for that path function. An ACS example is path;

```
path (A, Turbine A, C, H, I, H, Vapor Trap, Molecular Sieve,
      III, F, Ar/Cell Heat Exchanger).
```

If comparison is not acceptable with respect to both objects, then the path function is assumed to be a sneak function and is analyzed as such.

Path function as represented in hierarchical form is analyzed as a sneak function if it is not a validated path function. Sneak function determines if the path function is an enhancement or detriment to the

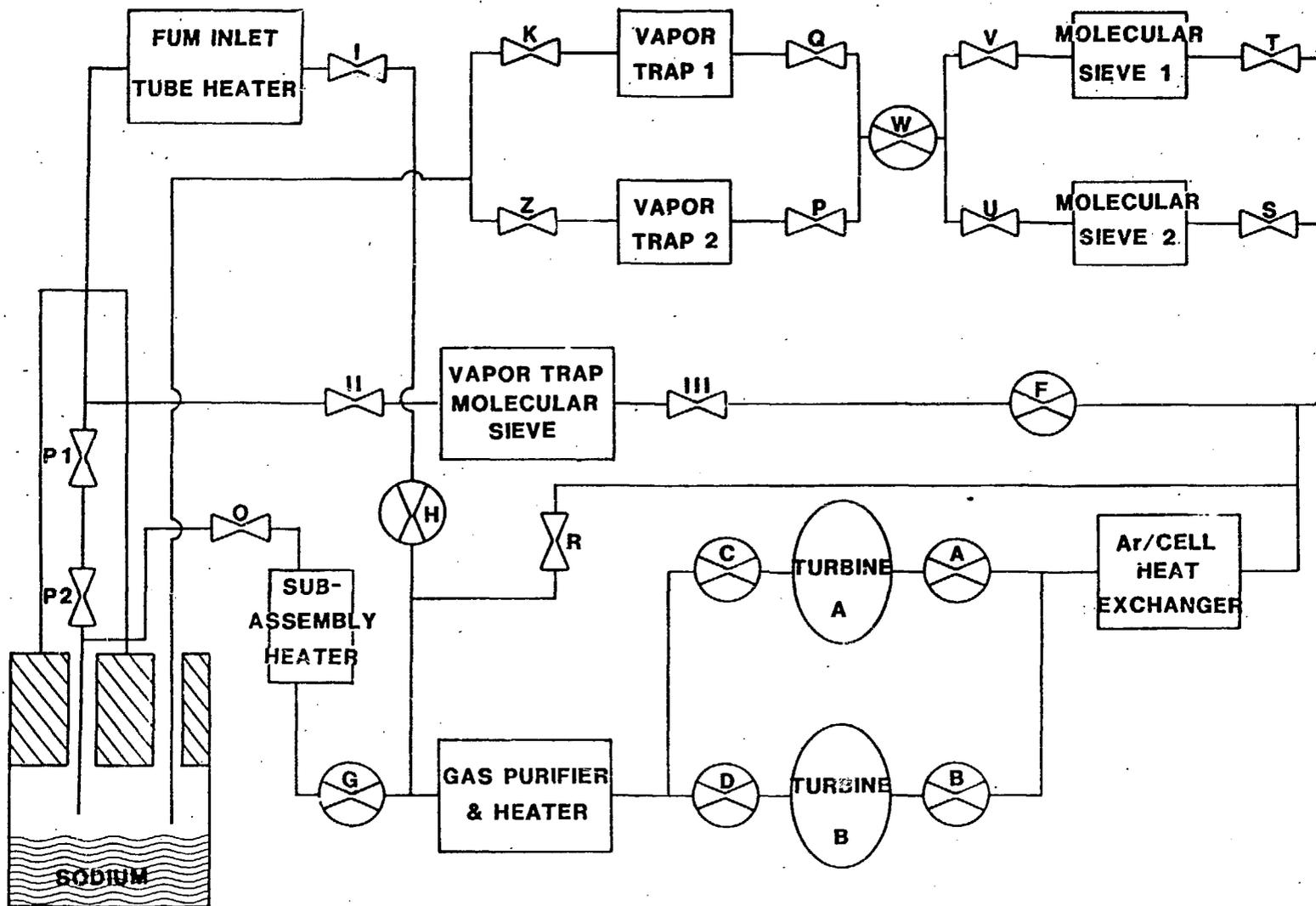


FIGURE 5: ARC CONFIGURATION OF THE ACS

system. If the sneak function matches a design function and is not an engineered design function then this path function is labeled as an enhancement. If the sneak function does not match a design function for the process, but matches a design function in the design function library, then it is labeled as a potentially enhancing sneak function. All other cases cause the sneak function to be labeled as a detriment. All analyzed sneak functions are output to the engineer for further analysis. Examples of enhancing sneak functions for the ACS are;

path (B, Turbine B, D, G, O, P2, P1, II, Molecular Sieve, Vapor Trap, III, F, Ar/Cell Heat Exchanger)

path (B, Turbine B, D, G, O, Z, Vapor Trap 2, P, W, V, Molecular Sieve 1, T, Ar/Cell Heat Exchanger)

Both paths provide cooling to the subassembly when the subassembly is out of the reactor tank and neither of the paths are described in any of the operating procedures. An example of a detriment sneak function respective to cooling is;

path (B, Turbine B, D, H, I, II, Vapor Trap, Molecular Sieve, III, F, Ar/Cell Heat Exchanger, R)

### CONCLUSION

Given the necessary relationships and definitions of design functions and components, validation of system incarnation and sneak function analysis can be achieved via automated reasoners. The relationships and definitions must define the design specification and incarnation functionally. For the design specification, the hierarchical functional representation is based on physics and engineering principles and bounded by design objectives and constraints. The relationships and definitions of the design incarnation are manifested as element functional definitions, state relationship to functions, functional relationship to direction, element connectivity, and functional hierarchical configuration.

A design function of the ACS is to maintain the temperature of a reactor subassembly below a specified temperature. This design function then evolves into the design specification and is expressed as functions that incorporate engineering principles, physics laws, objectives and constraints. The cooling design specification is defined as heat transfer surface, temperature gradient, heat-absorbing, heat-rejecting, and media motive force functions. The physical incarnation of the functions (the physical product of design) are then manifested, for example, as fuel clad and heat exchanger tubes, fuel, media and cell atmosphere temperature differential, argon gas and cell atmosphere, and turbines A and B.

Validation is performed by determining that the design incarnation functionally complies with the properties of the design specification. Sneak function analysis is the reasoning about an incarnation to determine unplanned element functions resultant from the incarnation. Validation

and sneak function analysis methodology involves the determination and operation of paths, path functions, and design functions.

Path is determined by the physical incarnation of the design specification via its representation as a graph constructed of vertices and component arc segments. Path function is then the sum of the arc functions in the path as defined by the component arcs and their relation to one another and flow direction. Path function is evaluated and represented in a hierarchical functional construct that differentiates between design and technical functions. This hierarchical configuration is then compared against the design specification to determine validation and sneak functions.

The relations, definitions, and methodologies developed in this project potentially have wide application in addition to validation and sneak function analysis. These relations, definitions, and methodologies can be applied to design, operation, and training. Transformation of a design concept for a process into a design specification is defined and bounded by physics laws, engineering principles, objectives, constraints, and technology. When presented in a functional hierarchical construct, the functions tend to layer themselves from top to bottom in the order of laws and principles, objectives, constraints, and technology. That is, the hierarchy tends to place the essential function toward the top and the physical functions towards the bottom. These hierarchies can be developed for specific processes by experts in that process field. The expert process hierarchy can then be utilized by less experienced engineers and consultant aids in future design of similar and same processes. These hierarchies can also be utilized to document the physical incarnation functionally and to analyze future design changes of the incarnated function. The hierarchy and path characteristics can be used to help develop normal and off-normal operations procedures for the process and can be used to analyze and process off-normal situations both diagnostically and prognostically. As a training aid, the functional hierarchy would explain and reinforce the functional aspects of the components and higher elements to the trainee. The path and pathfunction characteristics will help develop the trainee's ability to functionally analyze physical paths within the design incarnation, P&ID. That is, it will allow the trainee (either operator or engineer) to learn, review, and test the system concept of function and incarnation relationships.

Two other areas of potential application of these methodologies are fault tree analysis and alarm handling. Fault tree analysis requires the determination of all singular and integrated faults derived from electrical and mechanical designs. The design can be transformed into a graph consisting of arcs that represent the design elements, where each element has a defined number of states. Algorithms would then determine paths and path combinations associated with faulted elements states and conclude with a fault tree for the design in question. Alarms represent states of process element, i.e. systems, subsystems, and components. The matrix of alarms that result from a process fault is a function of the fault and the process elements associated with the fault. By defining the fault/component/function relationship, alarm states can be mapped into a graph representing alarm relations and hierarchy. These graphs can then be analyzed by "path" and "path function" algorithms to determine the hierarchical fault relationships during a giving alarm situation.

REFERENCES

1. Clocksin, W. F. and Mellish, C. S., "Programming In Prolog," ISBN3-540-11046-1 Springer-Verlag Berlin Heidelberg, New York, (1981).
2. Colley, R. W., "A Transition Control System for Procedure Prompting," 1982 ANS Summer Meeting, ISSN: 0003-018X, Vol. 41, p. 529, (1982).
3. Ehrlich, S. M. and Gabriel, J. R., "Cutsets With Required Arcs: A Prolog-Based Approach," ANL/MCS-TM-11, (1983).
4. Gabriel, J. R., "Algorithms for Automated Diagnosis of Faults in Physical Plant," ANL-83-70, (1983).
5. Kowalski, R., "Logic for Problem Solving," Elsevier North Holland, Inc. (1982).
6. Lusk, E. L., and Overbeek, R. A., "An LMA-Based Theorem Prover," ANL-82-75, (1982).
7. Seeman, S. E., Colley, R. W., and Stratton, R. C., "Optimization of the Man-Machine Interface for LMFBRs," Nuclear Safety, Vol. 24-4, pp. 506-512 (1983).
8. Stratton, R. C. and Lusk, E. L., "Automated Reasoning in Man/Machine Control Systems," 1983 ANS Winter Meeting, ISSN: 0003-018X, Vol. 45, P. 204 (1983).
9. Vaurio, J. K., et al, "Man-Machine Interface Program Plant," LMFBR Safety Program, Fast Reactor Safety Technology Management Center, Argonne National Laboratory, August 1982.
10. Wojcik, A. S. "Formal Design Verification of Digital Systems," 20th Design Automation Conference, 0738-100X/83/0000/022B, 1983 IEEE, p. 228-231 (1983).