17

Modular and Hybrid Expert System for Plant Asset Management

Mario Thron and Nico Suchold
ifak e.V. Magdeburg
Germany

1. Introduction

Industrial companies have to improve their production process as an important precondition for their survival on the market in a situation of a strong national and international concurrency. This includes a protection of the availability for use of the production equipment. This equipment includes simple devices like sensors and actuators but also complex machines and even more complex production lines and plants. The term "improvement" relates to the maximization of the production performance and a minimization of unintentional production down times.
This chapter contains an approach for the support of the plant asset management, which is an important part of the business management of a company. The plant asset management is focussed on the production equipment of an industrial company, while the general asset management involves all tangible and intangible assets of a company including buildings and patents. Furthermore plant asset management is mostly oriented to the operation phase of the production equipment. In the following we will take the term asset as synonym for production equipment.
Plant asset management has the goals to improve the reliability and efficiency of the assets and to reduce the effort to keep them in a state ready for production. Thus equipment maintenance is an important aspect of the plant asset management.
Important strategies of maintenance are:

- **Preventive maintenance**: The fitness to work of the assets is preserved by maintenance actions before any failure occurs. The action planning is based on production time and/or load cycles.
- **Predictive maintenance**: Maintenance actions are executed before any damages. The abrasion state of the assets is continuously monitored and logged here. The remaining useful life is predicted. This information builds the base for the planning of maintenance actions. This strategy is also called condition based maintenance. It leads to longer useful life time of assets compared to the preventive maintenance strategy.
- **Reactive maintenance**: Maintenance actions are executed after emerging failures of the assets. Repair actions or exchange of assets are necessary actions to take in most cases. This strategy has been used widely in former times when technical preconditions like computer aided planning systems and measurement systems where not available.

The reliability of production processes may be significantly increased by application of the preventive or predictive maintenance strategies in comparison with the application of the reactive maintenance strategy. However technical systems may and will fail. Thus it is not possible to completely pass on the reactive maintenance strategy.
The application of predictive and preventive maintenance strategies and economical constraints lead to a situation where the maintenance manpower is drastically reduced. The complexity of the technical systems grows in parallel for example by introduction of advanced automation systems and technical items dedicated to save energy and material during the production process. Thus in many cases a small group of maintenance employees is confronted with high complex systems in rarely but very critical situations, where production loss is caused by equipment failures.
Computerized assistance systems may support the maintenance employees at the completion of their tasks. This chapter describes an approach of such a system for the diagnosis of failure causes in complex production systems. The application of discrete theories is not sufficient for an optimal support of diagnosis tasks. Thus the approach contains the combination of various theories of the expert system technology and of probability theory. The reliability of detection of the right failure causes and the effort to formalize the necessary diagnosis knowledge are considered here as optimality criteria.

2. System Requirements

Expert systems for the diagnosis of technical systems are known since the 1960's and early 1970's. They have been successfully used for special systems. But they have not been applied for big, heterogeneous and complex systems. An analysis of end user requirements is necessary in order to find solutions for a broader appliance of such systems.
Figure 1 provides a summary of the use cases, which have been identified within the research project WISA (see acknowledgements in section 5). It is noteworthy that the diagnosis process is only one among a variety of use cases. A successful introduction of expert system technology depends a lot on the comfort for the creation of the knowledge base. Thus the consideration of the needs of other person groups is necessary beside the main focus on the maintenance personnel. Important user roles are:

- **Diagnostics user**. This is the user of the expert system. Mostly this will be a member of the maintenance staff. She is responsible for the availability of the assets.
- **Asset manager**. The asset manager is responsible for the efficient production. She controls the processes on business level. In most cases she is supervisor of the diagnostic user.
- **Asset expert**. The asset expert has special knowledge about the disturbance behaviour of the assets. This knowledge contains relations between failure causes and their symptoms. In many cases the expert is the constructor of the asset, but it is also likely that she is the asset user with long term experiences.

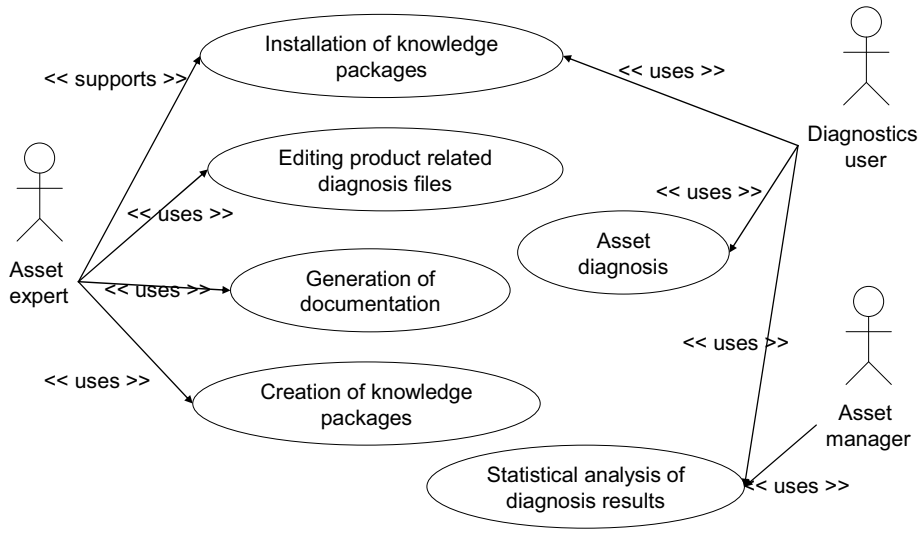

Fig. 1. Summary on use cases for a diagnosis expert system framework

The following list gives some details about the requirements resulting from the use case analysis:

- **Asset diagnosis**: This is the main use case. A failure event occurs and becomes visible for the diagnostic user who searches for the failure cause. The information about the failure may be distributed via industrial networks or may be a perception of the operation staff or the diagnosis user itself. Requirements are:
  - The input of the diagnosis user into diagnosis dialogs should be minimized.
  - If there are multiple failure causes possible then the list of causes should be sorted according to their probability.
  - The diagnosis system should ask back for additional symptoms, which make sense.
- **Editing product related diagnosis files**: The asset expert has to formalize his diagnosis knowledge. In most cases the asset expert is not an IT expert. Thus the input of the knowledge should be done not directly by using a programming language. Instead a forms based editor will be used. Requirements regarding such an editor are:
  - The graphical user interface of the editor should use maintenance terminology.
  - The editor should recognize and inform about syntactic and semantic errors within the knowledge base as far as possible.
- **Creation of knowledge packages**: The diagnosis knowledge of a complex asset includes the knowledge about the asset elements and the causal relations between them. A system is needed, which gathers all the knowledge, which contain the necessary information in order to create a knowledge package for the considered asset. Requirements regarding such a packaging system are:
  - The packaging system should gather all necessary knowledge in a file container.
  - The file container should be compressed for hard drive space efficiency.

- **Generation of documentation**: The equipment manufacturers should act as asset experts. Thus they investigate more time into the development of the products. The generation of a part of the necessary product documentation is a pay-back possibility. Requirements on a generator system are:
  - The documentation generator system should be included into the editor framework.
  - The generator system should only be parameterized by the asset main context. Necessary descriptions of asset elements will be included automatically.
  - The documentation result should be linked HTML pages, which may be transformed into other text processing formats.
- **Installation of knowledge packages**: The diagnosis expert system to be used by the diagnostics user will include the knowledge of a variety of products, which aggregate other components. Thus there is the need for installation of asset descriptions. Some requirements are:
  - The installation system should inform about type and version conflicts regarding descriptions of assets.
  - The installation system should contain features to gather knowledge packages via the Internet and from local resources.
- **Statistical analysis of diagnosis results**: If the failure behaviour of assets is tracked over a long time period, then there is the possibility to identify the assets, which are responsible for the highest production loss. Failure frequency and amount of production loss has to be tracked for this purpose. Requirements for this analysis are:
  - The statistical analysis system should identify the assets, which are responsible for the highest production loss.
  - The result presentation may be done in text form on-screen.

These system requirements should be satisfied by an intelligent language construction for the formalized asset descriptions and by a related software tool chain.

3. Design of the Knowledge Base

3.1 Special Criteria for the Knowledge Base

The introduction and the system requirements state, that the knowledge base will be constructed by multiple person groups. Thus there must be a way of collaboration between them. In the WISA project the following approach has been used:

1. A language has been designed for the formal description of diagnosis knowledge. It has been named **Hybrid Logic Description Language (HLD)** for purposes described below.
2. Tools have been designed for creation, transportation and installation of HLD files.
3. An HLD interpreter has been designed for the diagnostic process.

A variety of approaches are known to work fine for special diagnostic purposes. The problem is to select the right one(s) to meet industrial needs and strength. Thus a selection had to be made by matching against a catalogue of criteria for such a language. Some of these criteria are:

- **Processing of discrete causal relations should be possible.** That means that a specific symptomatic is uniquely related to a distinct failure cause. Such relations are often described in tabular form in conventional machinery and plant documentation.

- **Processing of vague diagnosis information should be possible.** Vague information here means such information, which don't may be acquired by measurement or only with inadequately hard effort. This kind of information is estimated by the diagnostics user mostly based on his/her visual, acoustical and tactile perceptions.
- **Processing of uncertain causal relations should be possible.** In opposite to the discrete causal relations it is often necessary to formalize relations between symptomatic and a variety of causes. These relations are weighted by different probabilities. Thus the language should provide constructs to formalize these kinds of relations.
- **The information processing concepts should be kept simple.** The concepts should be easy to understand for asset experts.
- **There should be the possibility to validate language instances.** The process of editing HLD files should be supported by checking them for syntactic and semantic errors.
- **There should be the possibility to re-use asset type based diagnosis knowledge.** The asset experts know the scope of the assets they describe, which includes all aggregates of the considered asset. The diagnosis knowledge about the aggregates should simply be referenced by the type and version of the aggregate in the description of the asset.
- **There should be efficient algorithms for the information processing concepts.** The knowledge base for a complex technical system will itself become very complex. The algorithm implementations should provide answers on the failure causes in an appropriate time (some seconds).

The following section describes the design of the HLD language, which has been developed to satisfy this list of criteria.

3.2 Hybrid Logic Description Language (HLD)

3.2.1 Propositional Logic and Its Representation as HLD Rules

The first means of expression is the propositional logic. Later on we want to introduce certainty factors. In order to motivate the introduction of these concepts let us have a look on the following example dialog between a machinery operator and an expert for the machinery:

Operator: "Production line 3 is currently out of order"
Expert: "How do the last products look like?"
Operator: "The surfaces are scratched."
Expert: "Did the forming machine made noise?"
Operator: "No."
Expert: "Check if there are crumbs in the raw material."
Operator: "Yes, there are crumbs."
Expert: "Then we have got low quality of the component X within the raw material.

This example shows a systematically diagnosis controlled by the expert. This expert has to be replaced by the expert system. Thus there is the need to formalize the shown diagnosis knowledge.

The kind of knowledge as mentioned in the example may be expressed by a rule base like the following:

(1) products_scratched IF line_3_not_working.
(2) tool_loose IF products_scratched AND forming_machine_noisy.
(3) crumbs_in_material IF products_scratched.

(4) component_X_bad_quality IF crumbs_in_material

These rules constitute only a small section of a diagnosis knowledge base for a real world application. The causes of symptoms are situated on the left side of the IF statement, while the symptoms itself are positioned on the right side. This is in opposite direction of the causal direction. The results of the application of the reasoning algorithms are the conclusions of the rules on the left-hand side of the IF statement and the result should be by definition the cause of symptoms.

The syntax of the propositional logic has been defined in various books like (Kreuzer and Kühling, 2006), (Russel and Norvig, 2003) or (Poole et al., 1998). Propositional formulae deal only with the truth values {TRUE, FALSE} and a small set of operations is defined including negation, conjunction, disjunction, implication and bi-conditional relations. The possibility to nest formulae enables arbitrary large formulae.

The HLD restricts the propositional logic to Horn-logic, which is not a big limitation. A Horn formula is a propositional formula in conjunctive normal form (a conjunction of disjunctions) in which each disjunction contains in maximum one positive literal. The set of elements of these disjunctions is also called a Horn clause. A set of Horn clauses build a logic program. If a Horn clause contains exactly one positive and at least one negative literal, then it is called a rule. The positive literal is the conclusion (or head) of the rule, while the negative literals constitute the condition part (or the body) of the rule. If a rule is part of a HLD file, then we call it a HLD rule.

The form of horn clauses is chosen for the HLD, since there exist an efficient reasoning algorithm for this kind of logic - namely the SLD resolution. This resolution algorithm may be combined with the breadth-first search (BFS) or with the depth-first search (DFS) strategy.

- **Breadth-first search**: The algorithm proves for each rule whether the conclusion is a consequence of the values of the conditions. Each condition value is either looked up in a variable value mapping table or it will be determined by consideration of rules, which have the same literal as conclusion. If there exist such rules, but a direct evaluation of their conclusion is not possible, then a reference to this rule is stored, but the algorithm proceeds with the next condition of the original rule. If there is no condition left in the original rule, then references are restored and the same algorithm as for the original rule is applied to the referenced rules. This approach needs a huge amount of memory.
- **Depth-first search**: This algorithm proves for each rule whether the conclusion is a consequence of the values of the conditions. Each condition value is looked up in a variable value mapping table or it will be determined by consideration of rules, which have the same literal as conclusion. If there exist such rules, but a direct evaluation of the conclusion is not possible then the first of these rules is evaluated directly. Therefore this algorithm does not need the references and saves a lot of memory compared to BFS.

It may be shown that the SLD resolution with BFS strategy is complete for Horn logic while the combination with DFS is incomplete. However, DFS is much more memory efficient than BFS and in practise it leads often very quickly to the result values. Thus both resolution algorithms have been prototypically implemented for evaluation of HLD files. The syntax of the HLD does not depend on the selection of search algorithms.

The propositional variables of HLD rules have special meanings for the diagnosis purposes. Following has been defined:

- **Symptoms** are propositional variables, which appear only as conditions within HLD rules.
- **Indirect failure cause**s are propositional variables, which appear as conclusion in some HLD rules and in other HLD rules condition part.
- **Direct failure causes** are propositional variables, which appear only as conclusions of HLD rules.

Thus simple propositional logic is modelled in the HLD by direct and indirect failure causes as conclusion of rules and by symptoms and indirect failure causes as conditions of rules.

3.2.2 HLD Rules with Empirical Uncertainty Factors

The application of HLD rules is not always applicable or at least not very comfortable because of the following reasons:

- A huge amount of rules and symptoms have to be defined in order to find failure causes in complex technical systems. This is accompanied by very large condition parts of the rules. The establishment of the knowledge base becomes too expensive.
- A diagnosis expert system, which has a high complex knowledge base, has to ask the users for a lot of symptoms in order to find a failure cause. Guided diagnosis becomes too time-consuming.
- Complex knowledge bases lead to long-term reasoning.

All these effects should be avoided according to the defined requirements. The mapping of simple cause-effect relations with simple HLD rules continues to be applicable. But complex circumstances need other kinds of expressivity.

A simple extension of HLD rules is the introduction of certainty factors (CF). Therein the conclusion of a rule is weighted with a certainty factor. Such systems are described for example in (Bratko, 2000), (Norvig, 1992) and (Janson, 1989). In these resources the value range for the certainty factors is the interval [-1, +1]. For a better comparability of the CFs with probabilities the interval [0, 1] has been chosen for certainty factors of HLD rules.

All propositions, which are evaluated by application of an algorithm on a HLD knowledge base, are weighted by a CF, since the conclusion parts of the rules are weighted by certainty factors. Certainty factors of propositions have the following semantic within HLD files:

CF = 0.0 The proposition is false.
CF = 0.5 It is unknown if the proposition is true or false.
CF = 1.0 The proposition is true.

CF values between 0.5 and 1.0 have the meaning that the related propositions are more likely true than false, while CF values between 0.5 and 0.0 mean, that the related propositions are more likely false than true.

Two algorithms for the evaluation of HLD rules with certainty factors have been tested. These are the simple evaluation algorithm according to (Janson, 1989) and the EMYCIN algorithm as shown in (Norvig, 1992). The simple algorithm is based on the following instructions:

1. The CF of the condition part of a rule is the minimum CF of all the conditions.
2. The CF of the conclusion of a rule is the CF of the condition part of this rule multiplied with the CF value for this rule.
3. If the knowledge base contains multiple rules with the same conclusion, then the CF of this conclusion is the maximum of the related CF values.

The algorithms for certainty factors are proved to provide incorrect results in some situations. On the other hand for MYCIN it has been shown that such systems may provide better results than human experts. In addition the rule CFs may be empirically determined and thus the creation of a knowledge base is very easy. For these reasons the concept of certainty factors has been included into the HLD language.

3.2.3 Fuzzy Logic as Part of the HLD

Rule sets as described in the previous sections use mappings of diagnosis relevant physical values to discrete values as propositions. Thus rules for each discrete value interval have to be provided. This leads to a big effort for the creation of the knowledge base. In this section we introduce Fuzzy Logic as one opportunity to improve the preciseness of the reasoning and to reduce the necessity for fine grained discretization levels of physical values. An example of a HLD fuzzy logic rule is the following:

motor_defect WITH 0.9 IF motor_windings_hot AND load_low.

The way of diagnosis is different from that of the propositional logic. The diagnosis user inputs values of continuous value spaces (in the example for motor winding temperature and mechanical load), instead of providing discrete symptoms and binary answering of questions. The result is again a value out of a continuous value space (in the example an estimation of the degree of abrasion of the motor). Special diagnosis relevant output variables have been defined for the HLD language.

The use of Fuzzy Logic for diagnosis purposes works in following steps:

1. Definition of the knowledge base: (A) Fuzzy variables have to be defined and (B) a Fuzzy rule set has to be integrated into the knowledge base.
2. Evaluation of the knowledge base: (C) The user inputs variable values and (D) the implementation of a Fuzzy Logic interpreter provides results by fuzzyfication of input variables, applying of inferences and by defuzzyfication of output variables.

Fuzzy variables may be defined by mapping of triangles, trapezoids or more round function shapes to terms of natural language. Input variables within the HLD fuzzy logic may be defined by piecewise linear membership functions, while output variables are defined by singletons (see figure 2).

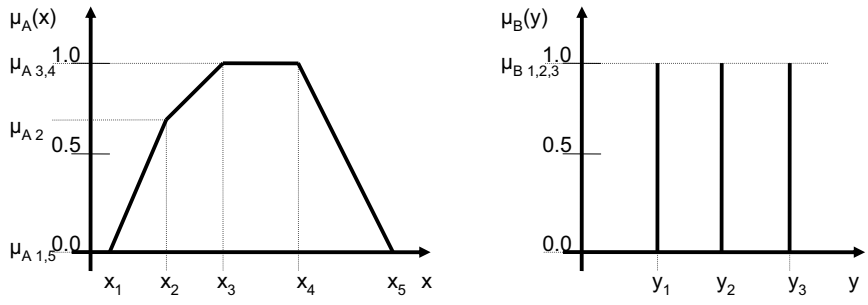

Fig. 2. HLD Fuzzy Logic input and output variables

This definition is in line with the standard (IEC61131-7, 1997). This is the standard for programming languages for programmable logic controllers (PLC). PLCs are the most used automation systems for machinery and plant control. Thus if the maintenance employees know something about Fuzzy Logic then it is very likely, that they know the terminology and semantics of this standard.

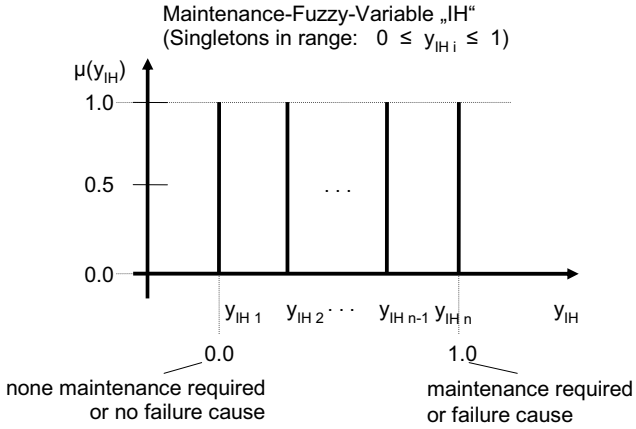

Fig. 3. HLD maintenance fuzzy variables

Beside the common semantics of Fuzzy output variables there are special definitions for maintenance variables in the HLD specification. This is illustrated in figure 3. The values of such variables y_{IH} are defined only within a range of [0, 1.0]. Only within this value range singletons may be defined. Similar to the definitions for certainty factors following conventions have been defined for these maintenance variables:

$y_{IH} = 0.0$ Maintenance is not necessary or this is not a failure cause.
$y_{IH} = 0.5$ It is not decidable if this is a failure cause.
$y_{IH} = 1.0$ Maintenance is necessary since this is a failure cause.

As mentioned above the processing of the maintenance knowledge base is done in three steps:

1. **Fuzzyfication**: Memberships are computed for each linguistic term of the input variables if there are numerical values available for the physical input variables.
2. **Inference**: The inference is done very similar to the approach used for rule sets with certainty factors:
    a. The membership of the condition part of a fuzzy rule is the minimum of all the memberships of the condition variables.
    b. The membership of the conclusion of a fuzzy rule is the membership of the condition part of this rule multiplied with the weighting factor for this rule.
    c. If the knowledge base contains multiple fuzzy rules with the same conclusion, then the membership of this conclusion is the maximum of the membership values of the conclusion variables.

3. **Defuzzyfication**: Within the basic level of conformance of the standard (IEC61131-7, 1997) the method "Center of Gravity for Singletons" (COGS) has been defined as defuzzyfication method. This has been taken over for the HLD specification. The result value of the fuzzy logic output variable is computed by evaluation of following formula:

$$y = \frac{\sum_{i=1}^{p} (\mu^{*}_{Bi} y_i)}{\sum_{i=1}^{p} \mu^{*}_{Bi}}$$

This formula uses the terminology as presented in figure 2. The μ^{*}_{Bi} are the membership values computed in the inference process for the p singletons at the values y_i. The result value y is the value of the output variable. Thus it is not a membership but a value of the value range defined for this output variable. Especially for the maintenance output variables the value range is [0, 1].

The approach of using singletons fits the need of fast computations as specified in the requirements analysis, since only multiplication and addition operations are used.

3.2.4 Bayesian Networks

Bayesian Networks have been introduced into the HLD, since the handling of uncertainty with certainty factors is not as mathematically correct as the probability theory does.

The example introduced in the propositional logic section could be extended by probabilities as follows

component_X_bad_quality (p=0.9) IF crumbs_in_material.
component_X_bad_quality (p=0.5) IF product_color_grey.

This example has the meaning that if there are crumbs in the raw material then the probability are very high (90%) that the material component X has not a good quality. In other words there are not many other reasons for crumbs than a bad material X. But there is another phenomenon in that approach: the variables crumbs_in_material and product_color_grey are not independent from each other. If there are crumbs in the material, then it is likely that the component X has a bad quality, but then there is also a good chance that the product looks a little bit grey.

Bayesian Networks are graphical representations (directed acyclic graphs) of such rules as shown in the example. (Ertel, 2008) gives a good introduction to Bayesian Networks based on (Jensen, 2001). One of the earlier literature references is (Pearl, 1988). There are following principles of reasoning in Bayesian Networks:

- **Naive computations of Bayesian Networks**. This algorithm computes the probabilities for every node of the network. The computation is simple but very inefficient. (Bratko, 2000) presents an implementation of this algorithm for illustration of the principles.
- **Clustering algorithms for Bayesian Networks**. This approach uses special properties of Bayesian Networks (d-Separation) for dividing the network into smaller pieces (clusters). Each of the clusters may be separately computed. For each cluster it is decided if it is influenced by evident variables. The computation of probabilities is done only for these clusters. The approach is much more efficient than the naive approach.

- **Approximation of Bayesian Networks**. Algorithms of this concept estimate the probability of variables. Such algorithms may be used even in cases where clustering algorithms need too much time.

The naive algorithm has been implemented for the evaluation of the usability of Bayesian Networks for the HLD. Further evaluation has been done by using the SMILE reasoning engine for graphical probabilistic models contributed by the Decision Systems Laboratory of the University Pittsburgh (http://dsl.sis.pitt.edu).

3.2.5 Summary and the HLD Language Schema

XML has been chosen as basic format of the HLD. Thus an XML schema according to W3C standards has been developed, which contains language constructs for the methodologies described in the previous sections. The structure of this schema is shown in figure 4.

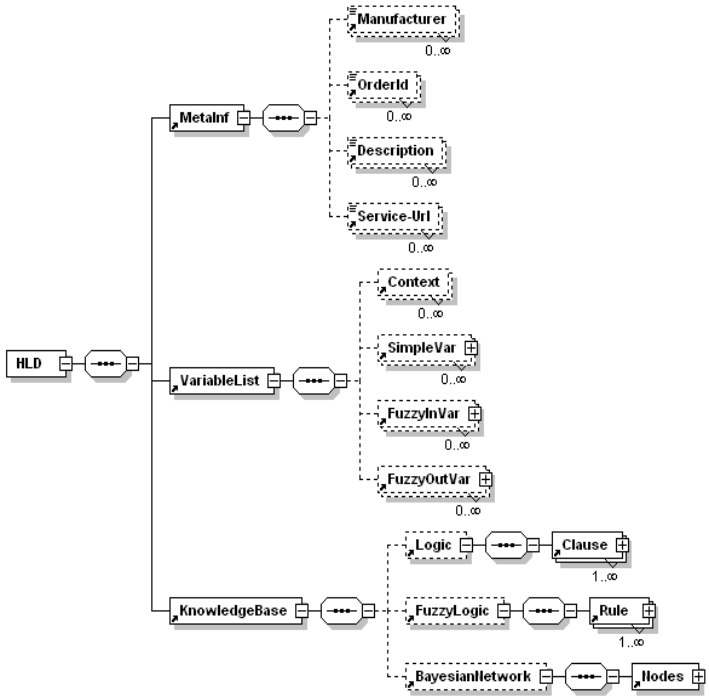

Fig. 4. HLD schema overview.

The HLD schema contains following top level information:

- Meta Information. The element MetaInf contains various common information about the asset described by the HLD file. This includes for example the manufacturer name, an ordering number, a short description and a service URL for getting further information from the manufacturer.

- Variable Declarations. The element VariableList contains lists of variables. Propositional variables (with and without certainty factors) are separated from Fuzzy Logic input and output variables due to their different representation models.
- Knowledge Base. This element contains the following sub elements:
  - Logic: This element contains rules with and without the use of certainty factors.
  - Fuzzy Logic: This element contains fuzzy logic rules and it references the Fuzzy Logic input and output variables.
  - Bayesian Network: This element contains the definition of a Bayesian Network for discrete variables. It contains conditional probability tables and references to the declarations of propositional variables.

The other attributes and elements define the semantics as specified in the sections above. The full HLD scheme may be downloaded at "http://i2service.ifak.eu/wisa/".

4. Framework for the Handling of the Knowledge Base

The central application of the HLD framework is the diagnosis system. It is implemented as a web application. This provides the possibilities to:

- maintain the knowledge base on one place,
- enable the access to the diagnosis system from any place,
- reduce the necessity of installation of special software (a Web browser is expected to be installed on any modern operating system by default).

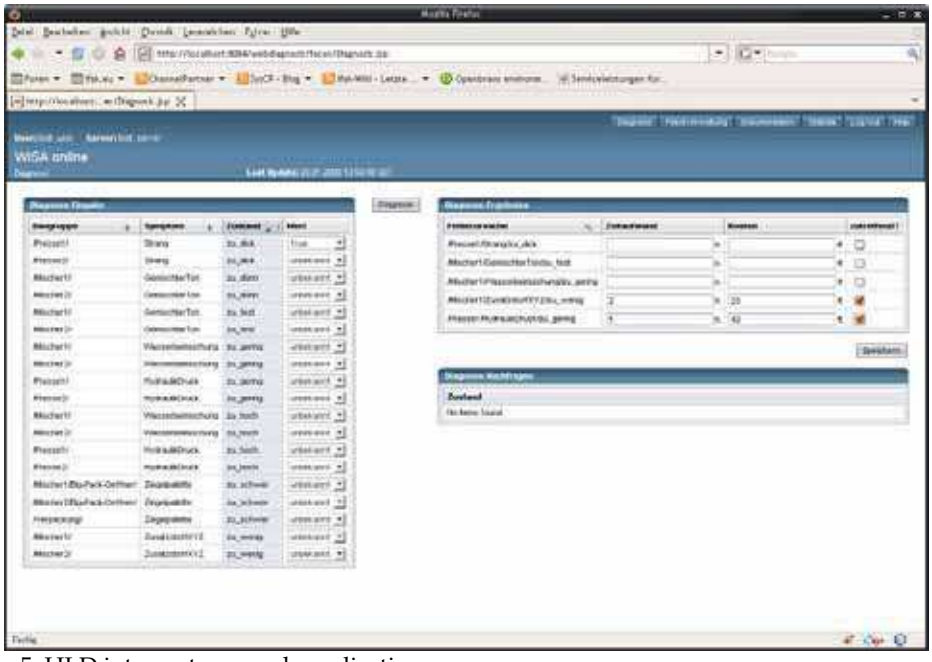

Fig. 5. HLD interpreter as web application

Figure 5 gives an overview of this application. On the left side the expert system provides a list of possible symptoms. The diagnosis user marks, which symptoms he has percepted. The diagnosis results are listed on the right side, sorted by their probability or their membership to a maintenance fuzzy variable.
The expert has another application for the creation of the knowledge for a specific asset type. This is an editor for HLD files. A screenshot of a prototype application is shown in figure 6. On the left side there is a tree representing the asset hierarchy. Elements of this tree are set into relations by definition of rules. This is done by entering some input into the forms of the right side. The screenshot shows the forms for input of logic rules. The entry fields are labelled by using the maintenance terminology. Thus a transformation of the terminology of artificial intelligence terminology to the application domain is done by this user frontend for the asset experts. The HLD editor uses for example the term "failure cause" ('Schadensursache') instead of the term "conclusion" or "clause head".

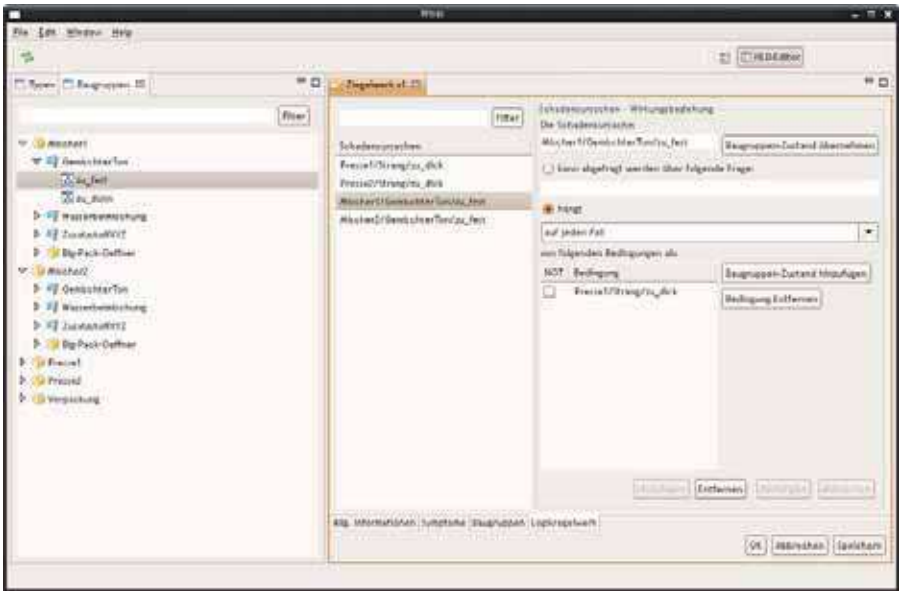

Fig. 6. HLD editor.

Assets like machine and plants are recursively nested when considering the aggregation relation. This is illustrated in fig. 7. If we consider a plant as asset, then the machines are the asset elements. If we further consider the machines as assets, then the tools, HMI elements and the control system are the asset elements. The HLD language introduces elements with the name "Context" in order to reference aggregated asset elements (see also fig. 4).

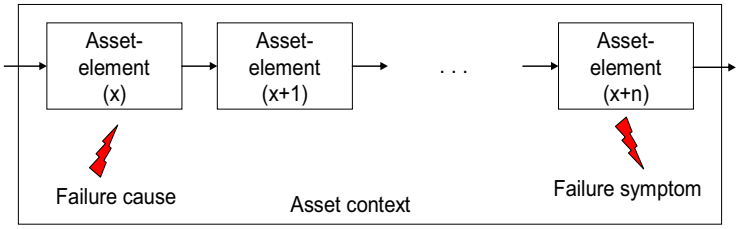

Fig. 7. Failure cause and symptom relation within an asset

In many cases failures occur in one asset element and cause symptoms in another asset element. These relations may be described in HLD files dedicated to the upper asset context, which contains the related asset elements directly or indirectly.
All HLD descriptions of the assets and their recursively nested aggregates build the knowledge base of the diagnosis expert system. They are positioned side by side in a HLD file repository. Each HLD description is dedicated to an asset type and its version, which are represented by structure elements of the repository. Thus the repository is not free form. It is obviously from fig. 7, that an asset description must be the assembly of the asset context description and the descriptions of all asset elements. Thus a HLD description is a package of HLD files with the same structure like a HLD repository. The tool set contains a packaging system, which assembles all necessary HLD descriptions from the repository of the asset expert and compresses them. Furthermore the tool set contains a package installation system, which decompresses the packages and installs them in a HLD repository, while paying attention to asset type and version information. In addition a documentation generation system has been set up, which generates HTML files out of a repository by a given asset context.

5. Conclusions and Future Research Work

An expert system has been introduced with a hybrid knowledge base in that sense that it uses multiple paradigms of the artificial intelligence research work. There was a gap between the success of the theoretical work and the acceptance in the industry. One key problem is the necessary effort for the creation of the knowledge base, which is overcome by the concept of a collaborative construction of the knowledge base by contributions of manufacturers of the production equipment.
Further research work will be spent to structure and parameter learning algorithms for the Bayesian Networks. The results have to be integrated into the HLD editor. Furthermore an on-line data acquisition will be integrated into the diagnosis system, which is especially necessary for an effective application of the Fuzzy Logic reasoning.
Most of the work has been done as part of the research project WISA. This project work has been funded by the German Ministry of Economy and Employment. It is registered under reg.-no. IW06215. The authors gratefully thank for this support by the German government.

6. References

Bratko, Ivan (2000). PROLOG - Programming for Artificial Intelligence, 3.Ed., Addison-Wesley

Bronstein, Semendjajew, Musiol, Mühlig (1997). Taschenbuch der Mathematik, 3. Ed., Verlag Harri Deutsch

Ertel, W. (2008). Grundkurs künstliche Intelligenz, 1. Ed., Vieweg Verlag

IEC61131-7 (1997). IEC 61131 - Programmable Logic Controllers, Part 7 - Fuzzy Control Programming, Committee Draft 1.0, International Electrotechnical Commission (IEC)

Janson, Alexander (1989). Expertensysteme und Turbo-Prolog, 1. Ed., Franzis Verlag GmbH München

Jensen, Finn V. (2001). Bayesian networks and decision graphs, Springer Verlag

Kreuzer, M.; Kühling, S. (2006). Logik für Informatiker, 1. Ed, Pearson Education Deutschland GmbH

Norvig, Peter (1992). Paradigms of Artificial Intelligence Programming - Case Studies in Lisp, 1. Ed., Morgan Kaufman Publishers, Inc.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann Publishers, Inc.

Poole, D.; Mackworth, A.; Goebel, R. (1998). Computational Intelligence - A Logical Approach, 1. Ed., Oxford University Press, Inc.

Russell, S. and Norvig, P. (2003). Artificial Intelligence - A Modern Approach , 2. Ed., Pearson Education, Inc.

Automation Control - Theory and Practice

Edited by A D Rodi

ISBN 978-953-307-039-1

Hard cover, 350 pages

Publisher InTech

Published online 01, December, 2009

Published in print edition December, 2009

The present edited book is a collection of 18 chapters written by internationally recognized experts and well-known professionals of the field. Chapters contribute to diverse facets of automation and control. The volume is organized in four parts according to the main subjects, regarding the recent advances in this field of engineering. The first thematic part of the book is devoted to automation. This includes solving of assembly line balancing problem and design of software architecture for cognitive assembling in production systems. The second part of the book concerns different aspects of modelling and control. This includes a study on modelling pollutant emission of diesel engine, development of a PLC program obtained from DEVS model, control networks for digital home, automatic control of temperature and flow in heat exchanger, and non-linear analysis and design of phase locked loops. The third part addresses issues of parameter estimation and filter design, including methods for parameters estimation, control and design of the wave digital filters. The fourth part presents new results in the intelligent control. This includes building a neural PDF strategy for hydroelectric satation simulator, intelligent network system for process control, neural generalized predictive control for industrial processes, intelligent system for forecasting, diagnosis and decision making based on neural networks and self-organizing maps, development of a smart semantic middleware for the Internet , development of appropriate AI methods in fault-tollerant control, building expert system in rotary railcar dumpers, expert system for plant asset management, and building of a image retrieval system in heterogeneous database. The content of this thematic book admirably reflects the complementary aspects of theory and practice which have taken place in the last years. Certainly, the content of this book will serve as a valuable overview of theoretical and practical methods in control and automation to those who deal with engineering and research in this field of activities.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Mario Thron and Nico Suchold (2009). Modular and Hybrid Expert System for Plant Asset Management, Automation Control - Theory and Practice, A D Rodi (Ed.), ISBN: 978-953-307-039-1, InTech, Available from: http://www.intechopen.com/books/automation-control-theory-and-practice/modular-and-hybrid-expert-system-for-plant-asset-management

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai

Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821