Managing Experience for Process Improvement in Manufacturing

Radhika Selvamani B., Deepak Khemani

A.I. & D.B. Lab, Dept. of Computer Science & Engineering I.I.T.Madras, India
khemani@iitm.ac.in
bradhika@peacock.iitm.ernet.in

Abstract.

Process changes in manufacturing are often done by trial and error, even when experienced domain personnel are involved. This is mainly due to the fact that in many domains the number of parameters involved is large and there exists only a partial understanding of interrelationships between them. This paper describes a framework for keeping track of process change experiments, before they qualify as full cases. Process changes happen as a result of diagnosis done by the expert, following which some therapy is decided. The paper also presents an algorithm for diagnosis and therapy based on induction on discrimination trees constructed on specific views on the set of problem parameters.

1 Introduction

Case Based Reasoning (CBR) has effectively been used in many domains to store and retrieve past cases to solve new problems. By storing and retrieving problem solving instances CBR is instrumental in exploiting experience in the simplest way. It is because of this simplicity that this cycle of CBR makes it a natural tool for knowledge management in various domains [1]. It has the ability to accrue a vast amount of experience in the form of problem solution pairs when it is embedded in any activity in which human solvers operate. But, by its nature, as a problem solving aid it is limited to reusing solutions from the past to similar problems. In such a scenario CBR system learn when human experts solve more problems. In this paper we explore ways in which the data available in case bases can be utilized to help the experts arrive at new solutions.

Storing and retrieving individual experiences is only the first stage of knowledge-based activity. There are many problems where the collective experience available with a problem solver plays a role in arriving at a solution. Traditionally the AI community has attempted to model abstract knowledge as rules directly. Such knowledge is acquired if a domain expert with a knowledge engineer can articulate the rules. Machine learning approaches investigate the automatic acquisition of knowledge in compact forms, or target functions [2]. Meanwhile, the data mining community has occupied itself with discovery of rules and associations from raw data. But often the goals of this exercise are loosely stated, since the objective is discovery of some interesting associations [3].

In this paper we describe a methodology of exploiting a repository of stored experiences, for specific goals. The objective of our induction algorithm is to identify a cause of a frequently occurring undesirable event, for example a particular kind of defect in manufacturing. Further, while adding this functionality to a CBR system in an industrial setting, we observe that different stakeholders in an organization make different kinds of knowledge demands. At one end a trainee is keen to learn what is routine, while at the other end the expert seeks to understand and classify what is out of place. This understanding often happens in stages, with the aid of repeated experiments with different ways of doing things. A knowledge management system should allow for such tentative knowledge to exist, even to be exploited, while it is being tested in the crucible of performance. In this paper we look at how experts explore the space of solutions, and how a case repository can maintain such tentative knowledge, and also the kind of support it can provide in arriving at informed solutions.

Section 2 describes the diagnosis problem in a manufacturing domain, and section 3 looks at experienced knowledge and its use in such a setting. Section 4 considers process improvement and section 5 describes an appropriate architecture to handle changing knowledge. Section 6 looks at how knowledge is used in the organization. Section 7 looks at an approach to diagnosis and repair followed by some results and concluding remarks.

2 Manufacturing Domain

We seek to extend an existing CBR implementation [4] in a refractory blocks manufacturing setting. Refractory blocks are manufactured by melting a mix of materials at high temperatures in an electrocast furnace, and pouring the melt into a mould. The blocks then need to be annealed to room temperature, and if everything has gone right, go through a finishing process before being inspected. The entire process may take a few months, and since the material and energy inputs are substantial too, there is a high premium on success. In the implementation described above, the CBR system acquires cases by data acquisition embedded in the shop floor. The process shops capture the information needed for a case as part of their routine [1], which becomes the case acquisition process of CBR systems. The quality control department detects problems in the final products and uses the retrieval cycle of CBR to get a solution from the case base. The CBR system retrieves a single case or a set of cases which matches the problem and process parameters of the final product.

2.1 Beyond Instance Based Solutions

There are situations when traditional CBR systems retrieving a single case for a given problem may not be able to provide a solution. Instance based CBR requires a successful case in its repository. In any manufacturing scenario process improvement exercises are periodically done to improve the quality of the products. New cases are created after some informed trial and error by domain experts. A knowledge management system could support this activity in two ways. One, by guiding the trial and error process with analysis of the past data. Particularly when the system may have large amount of knowledge captured and stored. Two, by allowing tentative process knowledge to exist while it is being tested, until a decisive case is at hand.

2.2 Defects in Manufacturing

Manufacturing processes are usually distributed among different shops and comprehensive expertise is not easy to acquire. Also the processes are prolonged involving many parameters. With a large number of parameters contributing to the product it is often difficult to pinpoint causes for defects [4]. In many ill understood domains, for instance foundries, manual analysis in the domain may be difficult and error prone. In such situations experts take recourse to trial and error methods to identify the cause of the problem. This brings in a need to integrate this problem solving activity into the CBR system. Also better diagnosis methods that aid experts would increase the usability and acceptability of the knowledge systems in the manufacturing domain.

3 CBR in Experience Management

CBR methodology originated from the cognitive field of human understanding and has been widely applied since late eighties. The Experience Factory approach gained importance during early nineties [5,6]. Experience management involves apart from knowledge storage and retrieval, sophisticated knowledge acquisition techniques, role identification exercise (of the stake holders), and dispersion of the knowledge widely referred to as “saying the right thing at the right time” [7]. Since the mid nineties CBR has been used both on the organizational experience factory process level as well as the technical experience base implementation level. Much of the work was focused on the software industries. [8, 9]. Knowledge management including experience management using CBR tools gained importance in different domains [10, 11]. Experience knowledge acquisition and usage in a manufacturing domain is different from other domains. A manufacturing domain requires any experience management solution to be embedded into the manufacturing process, rather than be developed as a stand-alone system for storage and dissemination of knowledge [12]. The CBR structures itself into the three main issues of knowledge management namely knowledge creation, knowledge integration and knowledge dissemination.

3.1 Knowledge Retrieval for Different Stake Holders

An instance-based implementation of CBR employs retrieval of past cases and adaptation of slightly different retrieved cases to solve the current problem. This makes the CBR attractive for training of novices and making day-to-day simple process decisions for the other employees. But it is of little or of no practical use for experts seeking to solve more complex problems. Since experts are involved in critical decisions regarding process improvement they tend to use CBR for just validation of their thoughts. At the same time if the experts were to ignore the CBR system the system in turn would be deprived of their experience. This concern for the issue of making CBR more useful to domain experts leads to the exploration of certain real problems faced by stakeholders in an organization. The following are the demands that different stakeholders may make on the system.

Requirements of the Trainees

- Browsing through the case base helps the trainees to learn about the domain
- Knowledge of how specific processes have evolved over time and the methods employed for design alterations would help understand why a particular process is being used.
- Knowledge of allowable process modifications *and* modifications that led to failure would facilitate informed modification / adaptation.

Requirements of the Case Author

- Knowledge about strength of cases. The more the case finds a match with a real problem, the greater its utility.
- Visualization of case matching. This would allow the author to validate the notion of similarity used by the retrieval system.
- Ease of maintenance of the system.

Requirements of the Domain Experts

- When there exist unsolved problems: For example focusing on a particular kind of defect an expert may look for some means of analysis of the past data to relate a set of process parameters to the defect.
- Experts need to be satisfied by the reasoning process involved in the analysis.

Requirements of the Design Department

- The process knowledge in the domain is not static. Changing environment and requirements often demand changes in the design process. In the absence of a precise domain theory, a CBR system could support a new process development.
- Statistical data stored with a case base could initiate changes in the design handbook.

In the next section we describe the process improvement scenario.

4 Process Improvement Cycle

Manufacturing poses problems continuously. In many domains it is more of an art rather than a science. There is a process cycle in the domain (Fig. 1). Applying the specified parameters for a particular process results in an outcome, which may be the defect in the manufactured product. The solution provided by the experts when the output is undesirable is usually a change in some of the process parameters. The changed process is applied in the domain to observe the outcome.
The process is revised until the desired outcome is obtained, a point at which the process is presumably flawless. Then, the process and the outcome form a complete case. In the intervening period, when the new problem is being experimented with, the experience instances remain as tentative cases.

4.1 Suitability of the Domain

The process cycle described above suits most domains where a trial and error change in the system is adopted to find remedies. The manufacturing domain has been chosen for our discussion. In this domain, parameters include the design specification of the product, but the diagnosis and alterations, which are discussed later in the paper, are applied only for the features involved in the production process, which can be modified affecting the outcome of the product. The defect description forms the outcome of the case. The alterations suggested for the production parameters in the trial case form the solution of the case. Since the complete case needs no further trial to be done, it does not require a solution part.

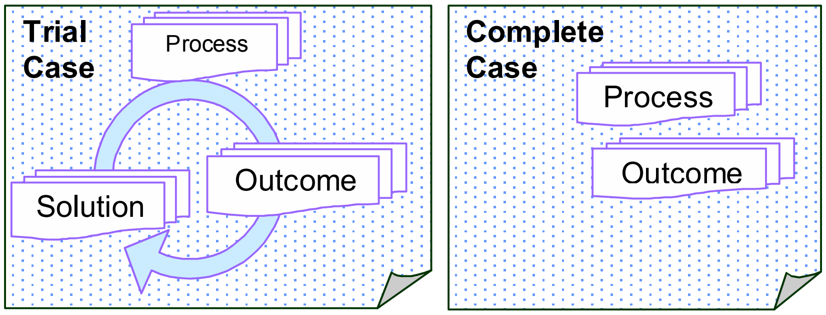


Fig. 1. Trial Case and a complete case in the process improvement cycle

5 Architecture for CBR in Process Improvement

5.1 CBR Module

The CBR module has two functionalities, case retrieval and diagnosis (Fig. 2). While retrieval takes place on flat structure of cases, diagnosis is done on hierarchical organization of a set of case instances [13, 14].

Case Retrieval. Conceptually CBR systems have a single case base used by the retrieval engine. But in the approach described here we have two different types of cases, trial cases and complete cases, stored respectively in a Trail Case Base (TCB) and a Complete Case Base (CCB). Any new case in the domain first enters the trial case base. When a process that consistently avoids the defects is found, the case graduates to the complete case base. A complete case is a case that has passed through the trials in the domain and has been finalized by the domain experts. Case history consists of all the instances belonging to the same trial in the TCB and the completed case in the CCB. It may be observed that retaining the instances, many of which have been classified under one case, is important for knowledge discovery tasks. A completed trail history will have cases both in the TCB as well as the CCB. A partial trial history may not have a case in the CCB. During retrieval for any new instance all the best matching cases are retrieved from both the TCB and the CCB.

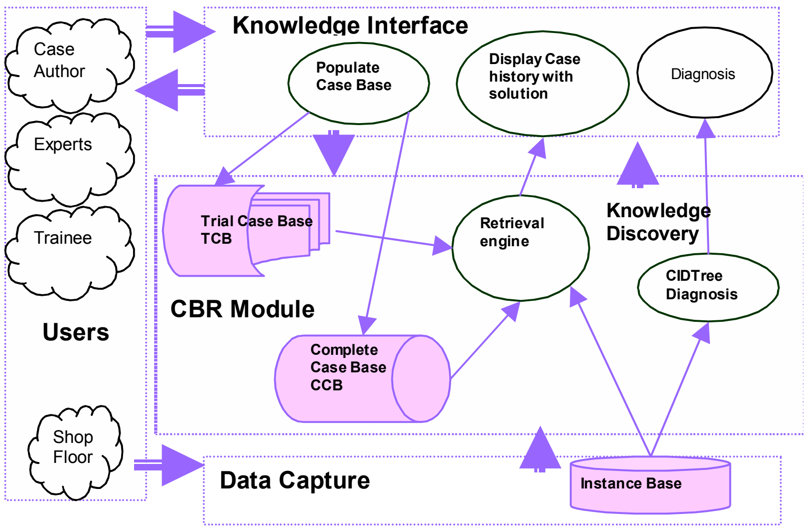


Fig. 2. Architecture for CBR in Process Improvement

Diagnosis. When a trial case has a bad outcome with sufficient number of instances, it needs improvement. Diagnosis process relates the outcome with the process parameters and therapy suggests alternative values for the parameters causing the defects. Our approach to diagnosis is described in section 7. Following diagnosis a new trial case is formed with the altered parameter values which is stored in the trial case base and the process is modified by the design department accordingly, for the products manufactured with the specified design in the case.

5.2 Knowledge Interface

The interface serves the purpose of capturing the knowledge from the experts as well as displaying the knowledge stored in the case base. It also lets users visualize the diagnosis process, so that they are convinced with the results. This is an extension over the information retrieval subsystem in the previous work [4]. A knowledge interface should cater to the different kinds of users for effective distribution of knowledge among stake holders. The various functionalities of the CBR module discussed above are useful for stakeholders at various levels in the domain. The shop floor personnel use the system for capturing and storing the data prevailing in the domain. The trainees, the domain experts and the technical staff involved in maintaining the system (case author) use the knowledge interface according to their needs. Access to the case history benefits the trainees who are new to the domain, enabling them to acquire the expertise of the manufacturing process. The domain experts on the other hand may use the system for analysis into the domain and solving new problems. Finally the maintenance of the cases within the case base is done by the case authors using the match values and the utility measures provided along with the cases during the retrieval. It is the case authors who define the case base, populating it with records from the instance base.

The data capture module taps into capture the data that prevails in the domain as a routine process, maps the data to required ontology and stores the formatted data in the instance base. Details about case acquisition and storage may be obtained from [4]. The instance base is used both by the CBR module and the Knowledge Interface. Provisions are provided in the knowledge interface to display the collected data which can be used to carry out further analysis using the CBR module. The diagnosis module in the CBR system uses the instance base to construct the memory structure needed for the investigation process.

6 Knowledge Flow

The requirements of the various stakeholders have been mentioned in section 3.1. Fig. 3 shows the flow of knowledge among various stakeholders. The instances of the newly manufactured products are provided to the users initially so that they can select the one, which is of their interest. The CBR module retrieves a set of case history matching the product instances, which may be of interest for new employees to know about the domain. The case author using the match value obtained during retrieval

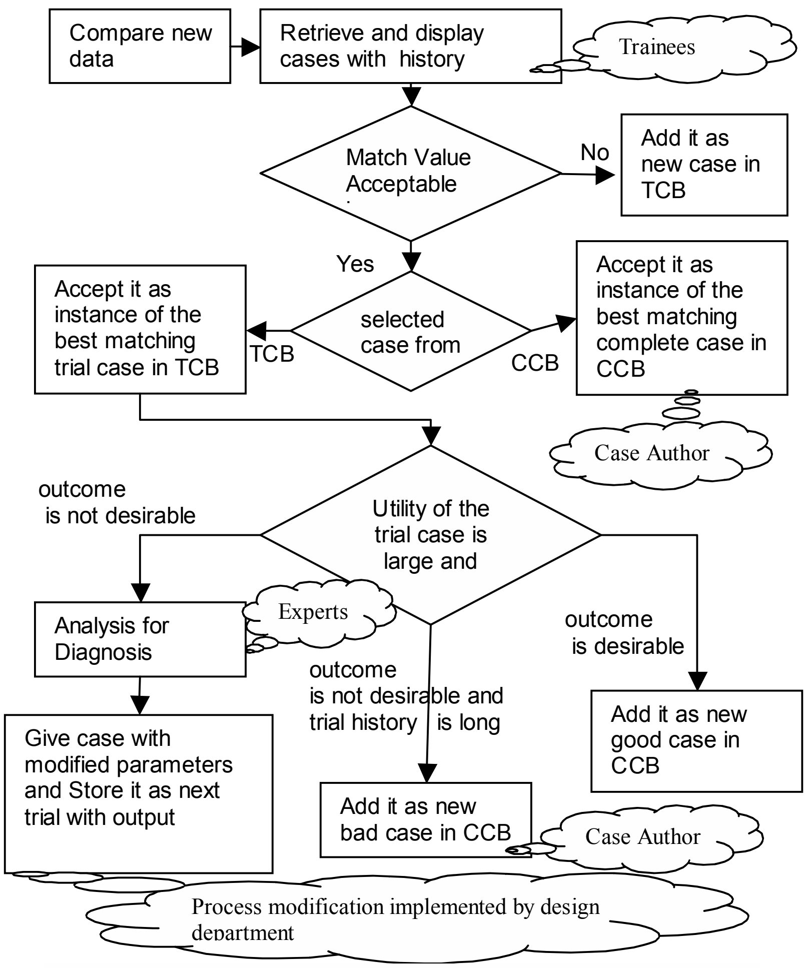

Compare new data
Retrieve and display cases with history
Trainees
Match Value Acceptable
No
Add it as new case in TCB
Yes
selected case from
TCB
CCB
Accept it as instance of the best matching trial case in TCB
Accept it as instance of the best matching complete case in CCB
Case Author
Utility of the trial case is large and
outcome is not desirable
Analysis for Diagnosis
Experts
outcome is desirable
outcome is not desirable and trial history is long
Give case with modified parameters and Store it as next trial with output
Add it as new bad case in CCB
Case Author
Add it as new good case in CCB
Process modification implemented by design department

Fig. 3. Knowledge Flow in the system

decides the case where the new data fits. If it does not fit anywhere, it is added as a new trial case. When they feel that the utility of a particular trial case is large and the outcome is not desirable, they can analyze the domain using the diagnosis and therapy module to get suggestions for process improvement.

The design department, which takes care of process modifications, needs to validates the suggestions using the cases in the case base and perhaps modify the design handbook. After looking at the trial history the design department can finalize the trials to form a complete case to populate the CCB. The CCB has both cases arrived at by a successful trial experiment and bad cases that mark the manufacturing processes that lead to undesirable outcome.

7 Diagnosis and Repair

We assume that defects can be traced to some parameter value setting. Looking at a case instance there would be no way of determining the parameter that is responsible for the defect. This is because a CBR system does not bring out any correlation or causal connections amongst the various attributes. However, looking at a significantly large set of defect cases it may be possible to isolate the culprit. We describe below a methodology that first builds a discrimination tree over a suspect set of parameters, and then uses a statistical measure to pick out the most likely fault choices. These parameters for example pertain to a particular sub process on the shop floor. We describe below an algorithm to do induction on discrimination trees constructed on specific views on the database. The system discovers plausible faulty choice of process parameters, and also suggests a therapy derived from successful past instances.

The previous work in the domain of aluminium casting discusses process improvement by trouble shooting and closing the loop of the manufacturing process [11]. It emphasizes the need for tracking the outcome of applying a case in the domain. Our work progressed with a goal to capture the outcome of a case and hence facilitating experiments in the existing domain processes. Data Mining techniques are often used to extract the required organization knowledge from the Experience base. [15]. We use classification trees for fault diagnosis using interestingness measures specially designed to focus the task.

7.1 CIDTree Methodology

The analysis for diagnosis is done over hierarchically organized cases. A methodology called CIDTree, cause induction in discrimination tree has been developed for diagnosis of the defect. Preliminary results based on the algorithm are described in [16]. The goal of building a CID Tree is to isolate those process parameters that could have led to the defect under consideration. Not only would one want to know the identity of the parameter, one would also like to determine the values of that parameter, which lead to the defect. The CID Tree is built as follows. First the user needs to identify the candidate set of attributes that are suspected. In our system this is done by selecting a view of the manufacturing process that focuses on a particular shop floor. Then the instances in the instance base are clustered to form two classes, good and bad, depending on the defect that needs to be analyzed. Next, the system uses c4.5 algorithm [17] to build a decision tree to segregate the defective cases from the non-defective ones. Ideally the algorithm would come up with a tree as shown in Fig.4(a). This tree has two leaf nodes, marked “Flawed” and “Suggestion” with zero entropy [17]. In practice however such a clean segregation may not occur and a larger tree with nodes containing a mix of defective and non-defective cases will be generated as in Fig.4(b). Here some parameter choices may be seen in the context of other parameter choices when the flawed node is deeper in the tree. The task now is to identify the most likely “flawed “ node. We use a measure described below to determine the most likely set of flawed nodes, that identify the process

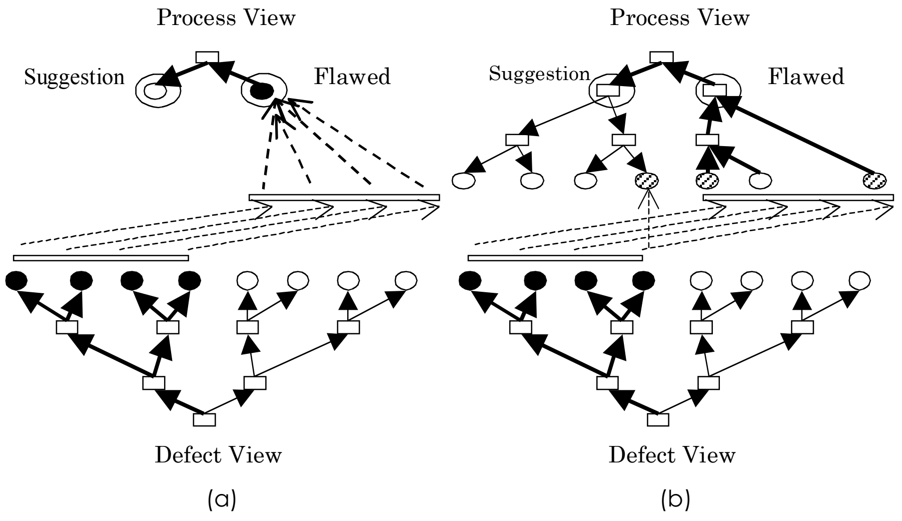


Fig. 4. Overlapping Clusters of the Process View with those of the Defect View Identifying the Flawed Node

parameter values that are likely to be the` cause of the defect. The sibling of the flaw node that is most successful becomes the suggested change in the parameter value.

Interestingness measure. Although specification of the task-relevant data and the kind of knowledge to be mined may substantially reduce the number of patterns to be generated, a data mining process may still generate a large number of patterns. Typically only a small fraction of these patterns will actually be of interest to the given user. A smaller set can be extracted by specifying interestingness measures or objective measures that estimate the simplicity, certainty, utility and novelty of patterns. The objective measures are based on the structure of patterns and the statistics underlying them [18]. Usually the interesting measure for a decision tree is its number of nodes or leaves. But we use this tree for functionality other than classification. Our interestingness measure calculation and the results are consequently different. After building the tree we classify all the instances under each decision node and determine the number and nature of the instances classified. We apply the objective measures to this tree structure to get the relevance score of the attributes with respect to defects. The relevance support is a measure of the attribute choice being related to the defect. Support and confidence are two widely used measures to determine the usefulness of an association rule mined from the database [18]. The interestingness measure for the association pattern (Current Node => Defect) in Fig. 7 is calculated as shown below.

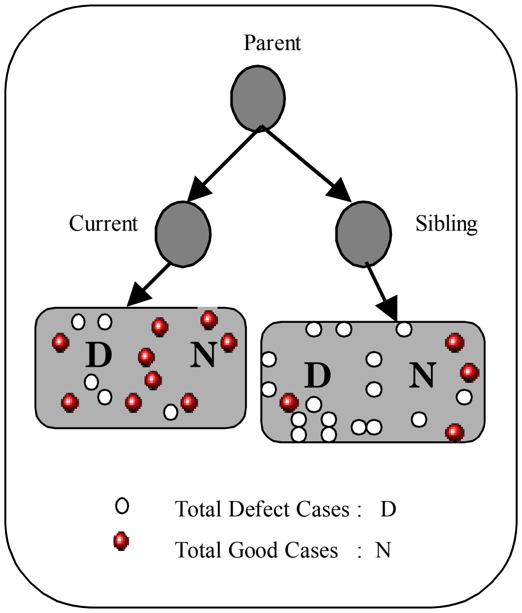


Fig. 7. Distribution of cases at the nodes and leaves

Support of an association pattern refers to the fraction of the total data tuples for which the pattern is true.

Support of (A=>B) = # (tuples with both A and B) / # (tuples in the data base) .
Where #(x) refers to number (x) .

Support (Current Node = > Defect)
=Fraction of defects under current node over all cases in the instance base
= $D_1/(N+D)$.

Confidence is the validity or trustworthiness of the pattern (ie) Fraction of defect cases out of all cases classified under current node.

Confidence of (A=>B) = # (tuples with both A & B) / # (tuples with A) .

Confidence (Current Node => Defect)
= Fraction of defect cases under current node over all cases classified under the current node
= $D_1/(N_1+D_1)$.

Maximize support and confidence (Current Node => Defective):

$$D_1^2 / (N+D) * (N_1+D_1 \quad . \tag{1}$$

Minimize support and confidence (Current Node => Non-Defective):

$$N_1^2 / (N+D) * (N_1+D_1) \ . \tag{2}$$

Minimize support and confidence (Sibling Node => Defective):

$$D_2^2 / (N+D) * (N_2+D_2) \ . \tag{3}$$

Maximize support and confidence (Sibling Node => Non-Defective):

$$N_2^2 / (N+D) * (N_2+D_2) \ . \tag{4}$$

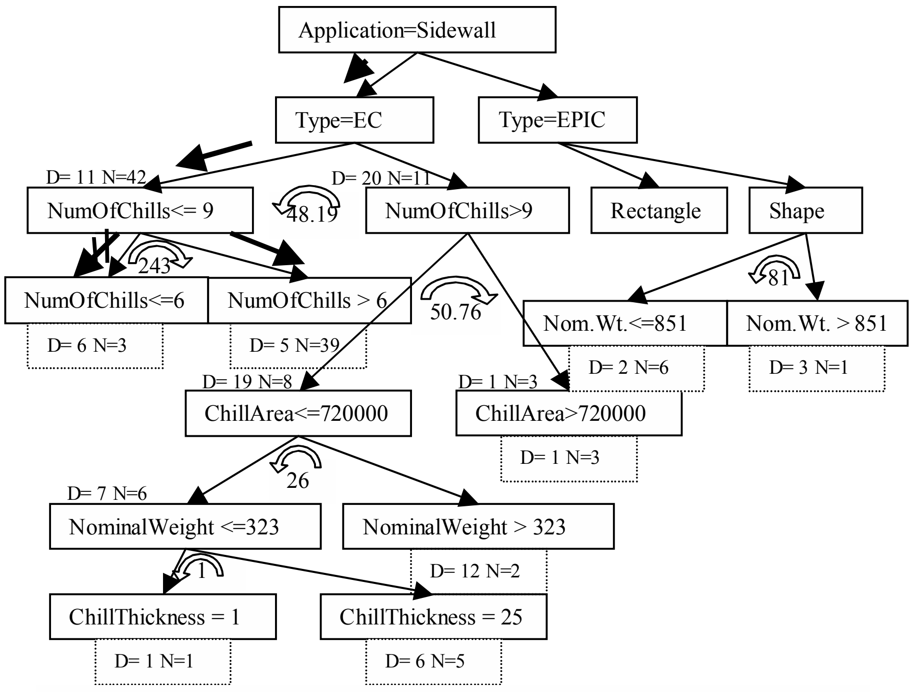
Relevance of Current Node in causing the defect

$$D_1^2N_2^2/D_2^2N_1^2 \ . \qquad (1 \text{ x } 4 / 2 \text{ x } 3)$$

Example. The diagnosis process has been tested in a domain manufacturing carborundum blocks [5]. There are two shops providing process parameters, the foundry and the furnace. The products manufactured have defects like cracks, spalls and hottear due to wrong parameter specifications. Process parameters for about 206 blocks manufactured in the domain was available in the instance base. There were 45 blocks. The instance base was manually clustered to defective and non-defective cases. Defective cluster had 38 instances and non-defective cluster had 168 instances. The induction tree was built for this training set of 206 instances. The number of defective and non-defective cases classified under each node is calculated as shown in Fig. 5. D denotes number of defective cases and N denotes number of non-defective cases. The relevance score is calculated for the nodes as shown in Table 1. The nodes 3 and 6 have high relevance with the defect and the nodes 4 and 5 are suggested respectively as the therapy. This can be seen visually in the Fig. 5. The arrow points towards the best alternative below each node. The relevance score is given below the arcs.

8 Conclusions and Future Work

The prime objective of the work described here is better process management. The Experience/Knowledge management tool is an application with a CBR system. We identify different needs of the different stakeholders. Two important requirements are to manage tentative process change experiments, and to provide decision-making support to experts to experiment with process changes. Cause identification for defects is done by induction over past experience and remedial actions are produced as a corollary. The cause is a flawed choice node, and the therapy is a (successful) sibling of the flawed node. The results of tests done in a few cases have been encouraging. In the new EM framework, the suggested therapy may be experimented with and stored as cases. Validation over a period of time will lead to its acceptance as a standard operating procedure. Future work is towards exploration of the diagnosis process using different induction algorithms



Diagnosis: If Type = EC and NumOfChills <= 9 then NumOfChills <=6 causes cracks.
Therapy : Allow numofchills > 6

Fig. 5. The discrimination tree and the induction process for diagnosis of cracks over the foundry parameters. D = Number of Defect cases. N = Number of good cases. The tail of the arc points towards the node relevant in causing defect and the head towards the alternative good node. The relevance scores are given below the arcs. The highest score is 243 and that gives the diagnosis and therapy. For score calculation refer Table 1

Table 1. Results tabulated on diagnosis of cracks over Foundry parameters. Possible Diagnosis (Node 3, 6) and suggested therapies (Node 4, 5) . The table was formed on the information derived at the nodes of a discrimination tree built over foundry parameters. D = defect cases classifieds under the node. N = Good cases classified under the node. Score 243 is the highest

Node No	Nodes	D Defect	N None	Score = $(D2^2*N1^2)/(N2^2*D1^2)$	1/Score	
1	NumOfChills<= 9	11	42	48.19		
2	NumOfChills > 9	20	11			
3	NumOfChills<= 6	6	3		243	Diagnosis
4	NumOfChills > 6	5	39			Therapy
5	Nom. Wt. <= 851	2	6	81		Therapy
6	Nom. Wt. > 851	3	1			Diagnosis
7	ChillArea<=720000	19	8		50.76	
8	ChillArea >720000	1	3			
9	NominalWeight <=323	7	6	26		
10	NominalWeight >323	12	2			
11	ChillThickness = 1	1	1	1		
12	ChillThickness = 25	6	5			

References

1. Watson, I.: Applying Case-Based Reasoning: Techniques for Enterprise Systems. Morgan Kauffman Publishers, Inc. San Francisco, CA , (1997) 61-77
2. Mitchell, Tom M.: Machine Learning. McGraw-Hill Companies, Inc., Singapore (1997)
3. Pazzani, Michael J.: Knowledge discovery from data ?, IEEE Intelligent Systems and their applications, Special issue on Data Mining II, Vol. 15, No. 2, March/April 2000, pp. 10-13
4. Khemani, D., Selvamani, Radhika B., Dhar, Anando R., Michael, S.M.: InfoFrax : CBR in Fused Cast Refractory Manufacture. in S. Craw, A. Preece (Eds.): Advances in Case-Based Reasoning, 6th European Conference, ECCBR 2002, Aberdeen, Scotland, UK,. Proceedings, LNAI 2416, (2002) 560-574
5. Althoff, K.-D., Decker, B., Hartkopf, S., Jedlitschka, A., Nick, M., Rech, J.: Experience Management: The Fraunhofer IESE Experience Factory, P. Perner (eds).: In Proc. Industrial Conference Data Mining. July (2001) Leipzig 24.-25
6. Birk, A., Surmann, D., Althoff, K.-D.: Applications of Knowledge Acquisition in Experimental Software Engineering. Fensel, D., Studer, R.: (Eds.): Knowledge Acquisition, Modeling, and Management (EKAW'99), Springer Verlag, (1999) 67-84.

7. Fisher, G.,Ostwald J.: Knowledge Management: Problems, Promises, Realities, and Challenges. In: Special issue on Knowledge Management. IEEE Intelligent Systems, Vol. 16, no. 1, Jan./Feb. (2001) 60-72
8. Bergmann, R., Breen, S., Göker, M., Manago, M., Schumacher, J., Stahl, A., Tartarin, E., Wess, S., Wilke, W.: The INRECA-II Methodology for Building and Maintaining CBR Applications. (1998)
9. Althoff, K.-D., Becker-Kornstaedt, U., Decker, B., Klotz, A., Leopold, E., Rech, J., Voss, A.: The indiGo Project: Enhancement of Experience Management and Process Learning with Moderated Discourses. In P. Perner (Ed.): Data Mining in E-Commerce, Medicine and Knowledge Management, Springer Verlag, Lecture Notes in Computer Science. (2002).
10 Althoff, K.-D., Bomarius, F., Müller, W. & Nick, M.: Using a Case-Based Reasoning for Supporting Continuous Improvement Processes. In: P. Perner (ed.): Proc. German Workshop on Machine Learning, Technical Report. Institute for Image Processing and Applied Informatics, Leipzig, (1999).
11. Price, C ., Pegler, I S., Ratcliffe, M B., McManus, A.: From troubleshooting to process design: closing the manufacturing loop. In proc. 2nd International Conference, ICCBR-97, Providence, Rhode Island, USA, July 25-27, Lecture Notes in Computer Science 1266 Springer (1997)
12. Michael, S. M., Khemani, D.: Knowledge Management in Manufacturing Technology, In Proceedings of the International Conference on Enterprise Information System 2002, Ciudad Real,, Spain, Vol. I. (2002) 506-512
13. Schank, Roger C.: Dynamic Memory Revisited. Cambridge University Press, Cambridge, London, NewYork (1999)
14. Kolodner, J.: Case-Based Reasoning. Morgan Kaufmann Publishers,Inc., SanMateo, California (1993).
15. Patterson, D., Anand, S S., Dubitzky, W., Hughes, J G.: Towards Automated Case Knowledge Discovery in the M2 Case-Based Reasoning System. In Knowledge and Information Systems: An International Journal. Springer Verlag, Vol. 1 (1999) 61-82.
16. Selvamani, Radhika B., Khemani D.: Investigating Cause for Failures in Discrimination Tree from Multiple Views. In the proceedings of the International Conference of Knowledge Based Computer Systems, Mumbai (2002)
17. Quinlan, J.R., C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers,Inc., SanMateo, California (1993)
18. Han,. J., Nishio, S., Kawano, H., Wang, W.: Generalization based data mining in object oriented database using an object cube model. Data and Knowledge Engineering, Vol. 25 (1998) 55-97