Nomusa Dlodlo[1, 3], Lawrance Hunter[2, 3], Cyprian Cele[3], Roger Metelerkamp[3], Anton F. Botha[3]

[1]National University of Science and Technology, Box AC 939, Ascot, Bulawayo, Zimbabwe, E-mail: ndlodlo@nust.ac.zw, ndlodlo@csir.co.za

[2] Nelson Mandela Metropolitan University P.O. Box 7700 Port Elizabeth, South Africa 6003 E-mail: lawrance.hunter@nmmu.ac.za, lhunter@csir.co.za

[3]Council for Scientific and Industrial Research, National Fibre, Textile and Clothing Centre, Box 1124, Gomery Avenue, Summerstrand, Port Elizabeth, South Africa 6000, E-mail: ccele@csir.co.za, rmetler@csir.co.za, afbotha@csir.co.za

A Hybrid Expert Systems Architecture for Yarn Fault Diagnosis

Abstract
This article describes a hybrid expert system architecture to support yarn fault diagnosis. The system uses a combination of rule-based and case-based techniques to achieve the diagnosis. Rule-based systems handle problems with well-defined knowledge bases, which limits the flexibility of such systems. To overcome this inherent weakness of rule-based systems (RBS), case-based reasoning (CBR) has been adopted to improve the performance of the expert system by incorporating previous cases in the generation of new cases. The idea of this research is to use rules to generate a diagnosis on a fault and to use cases to handle exceptions to the rules. The cases are represented using an object-oriented approach to support abstraction, re-use and inheritance features.

Key words: *expert systems, knowledge-based systems, yarn faults, case-based reasoning, rule-based reasoning, textiles, artificial intelligence, object-oriented.*

Introduction

Integrating computer technology and artificial intelligence is a promising approach in the provision of competitive services in the textile industries. Artificial intelligence (AI) is a branch of computer science concerned with the design and implementation of programs which are capable of emulating human thinking skills, such as problem solving, visual perception and language understanding. It is concerned with the creation of computer programs to perform activities which, if performed by a person, would require intelligence. The field of AI includes the areas of natural language processing, robotics, machine vision and expert systems [7].

Expert systems are computer models of human expertise in a specific domain of work. They are capable of offering advice and decision-support related to specific problem-solving in a well-defined knowledge domain. An expert system acts like an expert consultant, asking for information, applying this information to the rules it has learned, and drawing conclusions [10 and 25].

The typical expert system receives input describing a problem in its field of expertise, and then uses its inferencing technique to extract appropriate information from its knowledge base to produce an answer, diagnosis or description of a solution. Such systems have been used to interpret medical test results, diagnose problems with cars and determine the causes of telephone line failures.

The aim of this study is to design a hybrid knowledge-based system, which is capable of making use of heuristic knowledge and previous cases to improve its performance. The idea is to use rule-based reasoning to generate a diagnosis for a yarn fault, and to use case-based reasoning to handle exceptions to the rules. A characteristic of a rule-based system is handling problems from a well-defined knowledge base that contains rules. It uses a deductive approach to come up with solutions from a set of rules. In an only partially-understood domain, this approach may become impractical. Therefore one approach for improving rule-based systems is to extend the rule set by integrating the rule-based system with a case-based reasoning system. A case-based reasoning system is used to solve problems by incorporating past experiences in a partially-understood domain.

Rule-based expert systems

The idea of rule-based systems is to represent a domain expert's knowledge in a form called rules [25]. In a typical rule-based expert system, a rule consists of several premises and a conclusion. If all the premises are true, then the conclusion is considered true. For example, a diagnosis is true only when the reasoning is classified as a particular diagnosis.

The components of a rule-based expert system include the knowledge base, inference engine, knowledge acquisition component, and explanation system as illustrated in Figure 1.

Knowledge base. The permanent knowledge of an expert system is stored in a knowledge base. It contains the information that the expert system uses to make decisions. This information presents expertise gained from top experts in the field. This knowledge comes in the form of facts and rules. Facts are minimal elements of the knowledge which must be identified before anything else. For example, 'A slub is a yarn fault' is a fact. Rules consist of 'if….then' statements, where a given set of conditions will lead to a specified set of results. If a condition is true then an action takes place. For example, "if the twist is high, then a snarl fault results."

Inferencing. The inference engine is a computer program that controls the execution, and uses rules to respond to a query and determine whether a suitable match can be found in the fact list, through backward or forward chaining. It determines which rules will be applied

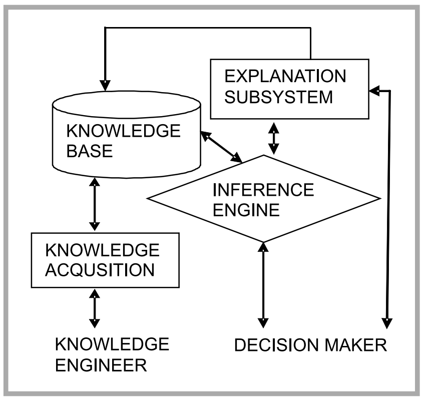

Figure 1. General structure of an expert system

to a given question and in what order. It uses information in the knowledge base. The inference engine drives the system by drawing an inference from relating user-supplied facts to a knowledge-based rule, and then proceeding to the next fact & rule combination.

Backward- and/or forward-chaining reasoning models are typically implemented in rule-based expert systems: Backward chaining is an approach that starts with a goal, e.g., "Which yarn fault is it?" and works through a potential thesis until it reaches the fact that supports the thesis.

Forward-chaining inference engines are goal-oriented in the sense that they try to prove a goal or rule conclusion by confirming the truth of all the premises. These premises may themselves be conclusions of other rules. This method begins with a set of known facts or attribute values, and applies these values to rules that use them in their premise.

Knowledge acquisition and explanation sub-system. Most expert systems continue to evolve over time. New rules can be added to the knowledge base by using the knowledge acquisition sub-system.

Explanation sub-system. Another unique feature of an expert system is its ability to explain its advice or recommendations, and even to justify why a certain action was recommended. The explanation and justification are done in a sub-system known as the justifier or explanation sub-system. It enables the sub-system to examine its own reasoning and explain its operations. The ability to trace responsibility for conclusions to their sources is crucial both in the transfer of expertise and in problem solving.

Case-based reasoning

According to [13], CBR is a problem solving technique which complements the solution, acting as a memory of past cases which can be consulted in order to identify similar cases for the new problem. CBR is described as a cyclical process comprising of the four 'Re's [1]. These are: (i) retrieve the most similar case, (ii) re-use the case to attempt to solve the problem, (iii) revise the proposed solution if necessary and (iv) retain the solution as part of the new case. CBR is an approach to incremental sustained learning, since a new experience is retained each time a problem has been solved. After identifying a given fault, a textile expert is reminded of another fault that he had come across some time ago. Assuming that the reminding was caused by a similarity of important characteristics, the textile expert uses the diagnosis and solution to the previous fault to determine the fault at hand and its solution.

A case is a contextualised piece of knowledge representing an experience. It contains a past lesson, that is, the content of the case and the context in which the lesson can be used [17]. Typically a case comprises:
- the problem,
- the solution to the problem, and
- the outcome after the solution has been applied to the problem.

In a case-based system, a problem is matched against cases in the case base, and one or more similar cases are retrieved. Case indexing [8] involves assigning indices to cases to facilitate their retrieval. A solution suggested by the matching cases is then reused. Unless the retrieved case is a close match, the solution will probably have to be revised, producing a new case that can be retained. For example if the past experience was a SLUB fault, and the fault that is current is a LONG SLUB fault, that means the case that is nearest to the new fault is the slub, and hence the revised content will have the characteristics of the slub as the base, as well as the additional characteristics that are peculiar to the long slub. Therefore inductive indexing is basically a search for similarities among a series of instance and categorisation based on these similarities.

The components of a case-based system are the input module, the case memory, retriever, the case adapter and a module to update cases.

Input module. The input module takes new cases. The case is forwarded to the indexing/matching module.

The case memory. The cases are stored in the case memory and may be represented as objects. The case memory is a hierarchy of objects. Any real world entity is uniformly modelled as an object. Every object is an entity that has a state, that is, the set of values for the attributes of the object; and a behaviour, that is, a set of methods which operate on the state of the object. The object-oriented paradigm is based on encapsulating the variables and the methods that operate on them into a single object. Conceptually, all interactions between an object and the rest of the system are via messages. Thus, the interface between an object and the rest of the system is defined by a set of allowed messages.

In CBR, the object attributes represent the internal characteristics of an object that facilitate the complete description of the case, and can be used to retrieve, reuse and interpret the case. The list includes attributes needed for classifying and indexing cases, attributes needed for interpreting the cases, attributes serving as the basis for decision-making and performance evaluation and attributes related to recognisable changes in scenarios that can be reused by others. Figure 2 is an example of an object on a yarn fault.

Retriever. In its most basic form, the case retrieval procedure consists of searching the case base to find historical cases that most closely resemble the current problem. The comparison consists of matching attributes of the current problem with those of each historical case.

Case adaptation. It is possible that the most similar case in the case base is significantly different from the current problem. Alternatively, there may be subtle but critical differences between them that invalidates the application of the old solution to the current problem. In such cases, it may be necessary to modify the historical solution to fit the current problem.

Case update. Once the current problem has been solved through the retrieval and adaptation of a historical case, the current case can be integrated into the case base as a new historical.

The advantages of case-based reasoning are as follows:
- The problem of the knowledge-elicitation bottleneck is surpassed, since

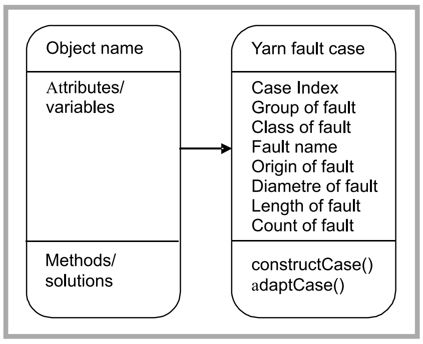

Figure 2. Structure of objects.

elicitation is a simpler way of acquiring past cases.
- Case-based reasoning systems can propose a solution quickly.
- Case-based reasoning systems can learn by acquiring new cases, making maintenance easier to demonstrate.
- Unlike rule-based systems, case-based systems can grow to reflect their organisation's experience by acquiring new episodic cases.

Object-oriented design

The object oriented paradigm is well-covered in literature [9 and 14]. Objects are autonomous entities that have a state and respond to messages. All objects communicate using the same mechanism of message passing, and the processing activity takes place inside the objects. Inheritance allows the classification of objects which permits their properties to be shared [39, 42, 43]. Abstraction in terms of object-oriented concepts is a technique that involves the selective examination of certain aspects of an application. It has the goal of isolating those aspects that are important for an understanding of the application, and suppressing those aspects that are irrelevant. Forming an abstraction in terms of classes and objects is one of the fundamental tenets of the object-oriented paradigm.

Re-using already available software components facilitates rapid software development and promotes the production of additional components. Taking components created by others is better than creating new ones. If a good library of re-usable components exists, browsing components to identify opportunities for re-use should take precedence over writing new ones from scratch. Inheritance is an object-oriented technique that boosts re-usability [24, 35].

Related work

Recently many publications have covered the development of knowledge-based systems using case-based reasoning in the areas of conceptual design (Lee, 1999), aircraft conflict resolution [16], military decision support systems [31], helpdesk operations [11], customer service management [12], legal systems [2, 4], diagnosis [22], design [21, 45], and planning [20]. It is seen that the applications of CBR in developing knowledge-based systems have been widely adopted in various industries and areas. There are a number of expert systems that have been developed using rule-based reasoning. This covers areas such as mining [27], commerce [28], diagnosis [37], medicine [50], and robotics [47].

In textiles, rule-based expert systems have been applied in determining dyeing recipes [15, 23], for the selection of fluorescent whiteners [3], for three-dimensional computer-aided intelligent design of garments [34], for fabric engineering [6, 36] and the analysis of defects in textiles [44].

Expert systems may incorporate other knowledge representation methods besides rule-based and case-based reasoning, such as frames, semantic nets, neural networks and fuzzy logic.

In textiles, neural networks have been applied in the identification of fabric defects [26 and 46], in the prediction of garment drape [19] fabric engineering [18] and the classification of spliced wool combed yarn joints [30].

Fuzzy logic has been applied to an intelligent diagnosis system for fabric inspection [32], while Bayesian networks have been applied an intelligent tutoring system in textiles [33].

Yarn faults

This section describes yarn faults, their origins and solutions [49]. This information is used in the design of the expert system.

GROUP 1- Faults with a characteristic length

Class A faults - Length about 4 mm

- NEP. A nep is a small accumulation of entangled fibres with a well-defined core. The diameter of the core is generally of the order of 0.8 mm to 1.5 mm. In general the total length of the nep is from 1.5 mm to 3 mm, and the local count 2 to 5 times that of the average yarn count. The origin of a nep is in the preparation stages, that is, at the top. It can be formed during the scouring of wool, opening and carding processes, and can be removed during combing. The problem could be that the roller beater in the blow-room may not be opening the fibres well enough to remove coils. During carding, 85% of neps should be removed under normal circumstances.
- SHORT FLY. The short fly is a mass of fibre of rather loose structure, generally loosely adhering to the yarn. The fault is most often shorter than 4 mm. The length of these short fibres does not fall within the range of the fibres being processed. It is easily distinguished from a nep by having a very loose structure, and by the fact that it has no compact core. They are loose because air suction and currents cause them to fly around. The solution is to improve the efficiency of such systems. A powerful extraction system must be kept in place to remove these fibres, ensuring minimal amounts of short fibres in the spinning area. Suction fans should be placed around the spinning area to remove such short fibres.
- KNOT. A knot can occur in the singles yarn, and also in the folded yarn. If the quality of the yarn as a result of preparation is poor, then knots may arise. Fibres must be straight. Drafting rollers need straight fibres, otherwise the front rollers result in a coiled fibre. If there is unevenness of sliver coming from the card and a variation in the evenness in spinning, thin regions as a result of short fibres occur.

Class B faults - Lengths between 4 mm and 40 mm

- WASTE. Waste is a compact mass of fibres analogous to a nep but distinctly larger in size having average adhesion to the yarn. The length of the fault is between 4 mm and 15 mm (more rarely up to 20 mm). The fault in the yarn originates from waste in the top. The faults are formed in the first stages of preparation and may disappear in drawing. Just like neps, the system must be able to extract unnecessary material such as short fibres, twigs, and seeds. Fibres would certainly form around any of these materials.
- FLY. A fly is a mass of fibre of rather loose structure, generally loosely adhering to the yarn. The fault is more often between 4 mm and 20 mm long but may reach 70 mm. The increase in local count, and particularly apparent diameter, may sometimes be very large, e.g. 2 to 20 times the average yarn. The increases and decreases in thickness at the beginning and end of the fault are not always abrupt. This fault is often formed by bundles of fibre drawn into the yarn during its passage to the winding machine / spinning frame. A powerful extraction system

to remove these fibres must be kept in place to ensure minimal amounts of short fibres in the spinning area. Suction fans should be placed around the spinning area to remove these.

- SLUB. A slub is part of a yarn with a thickness appreciably greater than the average over a fairly short length, of the order of 10 mm to 40 mm, and characterised by a fairly gradual appearance and disappearance of the thickened place. The local count will be between about 2 and 6 times the average. The corresponding part of the yarn is generally less or much less twisted. The fault is generally produced at the spinning frame by a faulty drafting action at certain times. The sliver then experiences a reduced amount of draft due to poor fibre control, releasing fibre bundles. The solution lies in how efficient the preparation has been, how well the system eliminates dust, and how well the fibres are paralleled.

Class C fault – Length between 40 and 160 mm

- LONG SLUB. Part of the yarn with a thickness appreciably greater than the average over a fairly long length ranging from 40 mm to 160 mm. The emergences and disappearances of the thickness are generally very gradual. The local count may be up to about 6 times the average. The corresponding part of the yarn has generally very little twist. The fault is produced in the spinning frame by a faulty drafting action, but is in general caused by more pronounced mechanical defects, or more pronounced faults in the roving.
- PIECEING UP. Every time an end breaks during spinning, the yarn end of the bobbin is located, withdrawn and rethreaded, bringing it into close proximity with the strand of fibres being delivered from the drafting zone, so that the twist binds the two ends of the yarn together where they overlap. This is called piecing up. Faults arise due to careless piecing at the spinning frame. When the yarn is pulled from the bobbin back to between the drafting rollers, it meets new fibres. The length is between 40 mm and 60 mm (exceptionally up to 200 mm), and the local count between 2 and 6 times greater.
- CRACKER. Refers to parts of yarn with a thickness clearly greater than the average, over a length most often between 40 mm and 60 mm, but different from the long slub fault by having a characteristic spiral appearance, some of the fibres comprising the yarn being wrapped around the other part in a corkscrew fashion. The local count is generally about 2 to 6 times the average; the emergence and disappearance of the thick part are fairly gradual. The fault occurs at the spinning frame as a consequence of the formation in the drafting zone of two parts of the roving which were drafted differently. The part drafted most winds itself round the part drafted least in the manner of a corkscrew. As in the slub, the problem could be that the covering rollers are be worn out at the spinning phase. This problem appears during roving and in the spinning system.

Class D faults– Length above 160 mm

- THICK YARN. This refers to part of the yarn with a thickness appreciably greater than the average (as in the slub), of the order of 160 mm to 1 metre in length (in rare cases up to 2 metres or more), for which the emergence and disappearance of the fault are generally very gradual. The local count is generally between 1.5 and 4 times the average. This fault is generally caused by a thickening similar to a slub present in the finisher roving, where it appears for reasons analogous to those indicated for yarn. Drafted from about 10 to 25 times at the spinning frame, this 'slub' produces long thick places in the yarn, at times up to about 2 metres in length.
- SPINNERS DOUBLE. Where an end breaks and piecing occurs, a fault is produced by the total or partial fusion of two ends at the spinning frame. Once inside the rollers and joining new fibres, the redundant fibre has to be cut out. If it is left too long, it doubles up because of new roving coming in the ring system. Local count is about 1.5 to 2 times greater than the average. The fault is often quite long, up to several metres.
- TWISTING DOUBLE OR TWISTING LASH-IN. Accidental fusion of two ends in twisting. This fault is caused by the winders. For twisting to take place, some winding should be done first. Instead of a single yarn in the bobbin going in after joining, the loose end may not be located. When it gets to the winder, it comes together with the other loose end, and they come together. Winding should ensure that this fault is eliminated as soon as it is detected. Local count is generally twice the average. The fault is often rather long.

,GROUP 2 - Faults of variable length

- DOUBLE THREAD OR WINDING LASH-IN. This is the length of yarn which escapes from the hand of the spinning operative (and is then spun by the twist in the yarn), or the winding operative (resting nearly freely on the yarn, with little binding twist), and proceeds to get entangled with the main part of the yarn. Local count is about double the average. The length generally is from 2 cm to 20 cm. The length of yarn which escapes from the hand of the spinning operative gets entangled with the main part of the yarn.
- SNARL. Occurs on yarns which are badly set and have a high twist. In folded yarn, it sometimes causes quite a bad fault by twisting itself around the yarn. Twist liveliness is the degree to which the yarn coils around itself. If the twist is too high, then twist liveliness is increased in an effort to improve the strength of the yarn.
- LOOP. The fault occurs at a time of a sharp breakage in winding due to insufficiently tensioning the yarn held in the hand when restarting after a breakage. The fault occurs predominantly with yarns which have a tendency to snarl.
- FOREIGN MATTER: BURR, STRAW, SHIVE, etc. Faults deriving in general from the top in the case of the wool.
- FIBRE RING OR RUB-UP. Fibre ring is an accumulation of fibres surrounding the yarn, forming a ring of greater or lesser length. This fault most often has a length from 4 mm to 10 mm, but may reach 20mm or more. The emergence and particularly the disappearance of the thick part are in general abrupt. The ring of fibres is distinguished from WASTE and SLUB faults by the fact that it is only lightly attached or is not attached at all to the yarn, and by the fact that it encircles the yarn. It can easily be made to slide along the yarn. The fibre can be produced at the traveller of a ring frame, or on various yarn guides during spinning and winding. The solution is to ensure that the system is efficient enough to get rid of excess

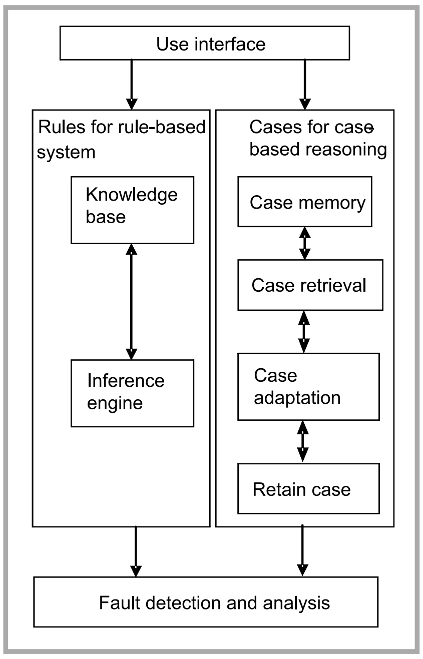

Figure 3. *Hybrid architecture for yarn fault analysis.*

matter. Suction units should be in place around the traveller to remove dirt during spinning.

The hybrid architecture

The architectural goal is to improve rule-based reasoning by augmenting it with case-based reasoning. This augmentation is done by taking the rules as a starting point of problem-solving, i.e. using rules to generate a first approximation to the diagnosis for a target fault and then invoking case-based reasoning to handle exceptions to the rules. The idea is to fine-tune the performance of the rules. Another advantage is that if the case-based reasoning misses a similar case, the architecture will at least have a reasonable default answer generated by the rule-based system. The architecture of the hybrid expert system is shown in Figure 3.

Each subsystem is essentially an individual problem solver. If a case-based solution is available, case-based reasoning can learn by acquiring new knowledge. Therefore the architecture should cater for this. The input uses fault symptoms and produces a casual network of possible internal states that lead to those problems. When a fault arises, the system tries to find faults with similar, but not identical, properties. The system adapts the diagnosis by considering the differences in symptoms between the old and new cases. The component to add a new case is generated.

In order to solve a problem about specific faults, baseline knowledge about specific issues is necessary; e.g., to solve LONG SLUB, one needs to have knowledge on SLUBS.

The flow in the system is as follows as shown in Figure 4:

The rule-based expert system components for yarn fault analysis

Classification using decision trees can be used to extract models describing important data classes or to predict future data trends. A decision tree is a flowchart-like structure where each internal node denotes a test on an attribute, each 'branch' represents an outcome of a test, and the 'leaf' nodes represents classes or class distributions. The decision trees can easily be converted to classification rules.

A decision tree provides a procedural guidance to the yarn fault problem as seen in Figure 5.

The facts in the knowledge base of the rule-based expert system are class A, group 1, known diameter, etc.

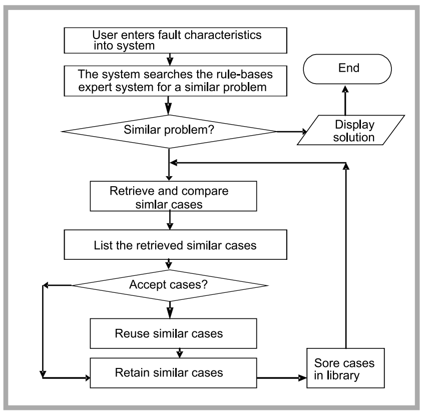
User enters fault characteristics into system
The system searches the rule-bases expert system for a similar problem
End
Similar problem?
Display solution
Retrieve and compare simlar cases
List the retrieved similar cases
Accept cases?
Reuse similar cases
Retain similar cases
Sore cases in library

Figure 4. *Flow of data in yarn fault analysis architecture.*

Knowledge is represented in the form of rules as follows:

IF group 1
 AND class A
 AND known diameter
 THEN NEP
IF group 1
 AND class A
 AND unknown diameter
 AND compact structure
 THEN KNOT

Questions such as the following should be answered at the user interface:

'What is the group?'
'What is the class?'
'Is the diameter known?' , etc.

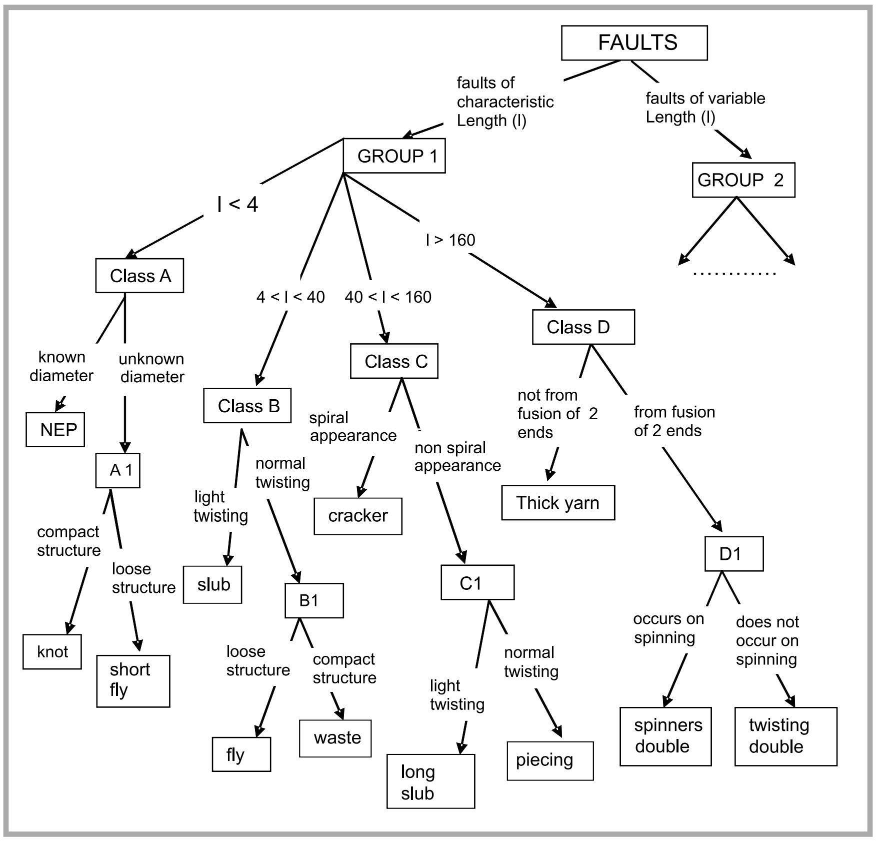

Figure 5. *Decision tree to yarn faults.*

The inference engine to the above decision tree takes the form:

```
Method Faults
{If (fault of characteristic length)
Call Method Group 1
Else Call Method Group 2}

Method Group 2
{If (length<=4)
Call Method Class A
Else If ((length>4)&&(length<=40))
Call Method Class B
Else If ((length>40)&&(length<=160))
Call Method Class C
Else Class D}

Method Class A
{If (fault of known diameter)
Fault name is NEP
Else Call Method A1}

Method A1
{If (fault has compact structure)
Fault name is KNOT
Else Fault name is SHORT FLY}
```

The case-based expert system components for yarn fault analysis

Let us consider the example of the SLUB and LONG SLUB faults. Except for the differences in the lengths of the faults, the rest of the characteristics are similar. Supposing that the rule-based expert system has captured only the SLUB fault, then if the fault detected proves to be a LONG SLUB, that would mean that its characteristics have to be captured in the case-based reasoning expert system. The LONG SLUB will inherit the characteristics of the SLUB, and in addition, the length is modified. To represent the inheritance in object-oriented terms:

In Figure 6, the rectangle represents the inheritance. This means that the LONG SLUB object will inherit all the characteristics of the SLUB fault and will additionally have a method to reset the length of the fault, resulting in the LONG SLUB.

```
Class SLUB
{int setGroup(itsGroup)
int setClass(itsClass)
string setFaultName(itsFaultName)
string setFaultOrigin(itsFaultOrigin)
int setFaultLength(itsFaultLength)
int setFaultCount(itsFaultCount)}

Class LONG SLUB : public SLUB
{public:additional characteristics()}

int main
{LONG SLUB ls
ls.setGroup(1)
ls.setClass(c)
ls.setFaultName(long slub)
ls.setFaultOrigin(faulty drafting in spinning
frame)
ls.setFaultLength(10-40 mm)
ls.setFaultCount(2-6 times average)
return 0}
```

The sub-program above shows a declaration of the object on the SLUB fault as a class with all its characteristics. The object LONG SLUB inherits the characteristics of the SLUB in addition to its own extra characteristics. The main subroutine calls the object LONG SLUB and assigns the actual values to create the object.

Benefits of architecture

The architecture has the advantage of being dynamic, that is, it allows the acquisition of new knowledge during the lifetime of the expert system. Other expert systems are static; that is, they are created once and for all. Experts gain new experiences and knowledge in life which they use, and this trend should be reflected in expert systems as well. The object-oriented approach ensures incremental development. New knowledge is added to the system as objects. The addition of one extra object does not mean that an expert system has to be rewritten from scratch.

Summary

In this paper, a hybrid rule- and case-based system has been proposed for improving the accuracy of the rule-based system through case-based reasoning. The idea is to generate an approximate answer to the problem using rule-based reasoning, and to use case models to handle exceptions to the rules and structural details.

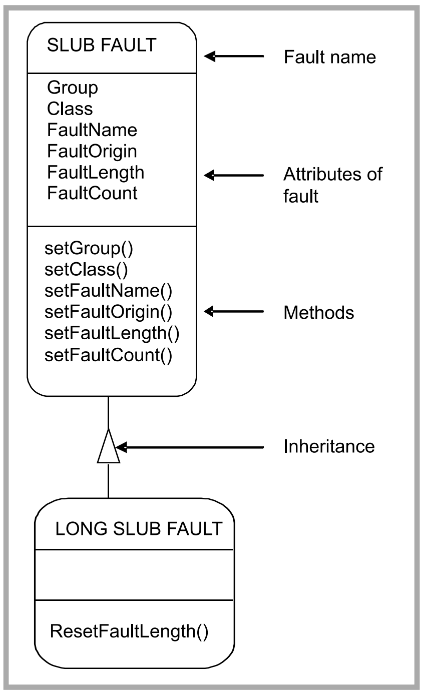

Figure 6. *Inheritance in objects.*

References

1. *Aamodt A., Plaza E., 'Case-based reasoning: foundational issues, methodological variations and system approaches', Artificial Intelligence communications (AI Com), Vol. 7, No. 1, p. 35-59, 1994.*
2. *Ashley K.D., 'Arguing by analogy in law: a case-based model'. D.H. Helaman (eds.), Analogical reasoning: perspectives of artificial intelligence, Cognitive Science and Philosophy. D. Redei, 1988.*
3. *Aspland J.R., Davis J.S., Waldrop T.A., 'An expert system for selection of fluorescent whiteners', Textile Chemist and Colorist, Vol. 23, No.9, p. 74-76, 1991.*
4. *Bain W.M., 'Case-based reasoning: a computer model of subjective assessment'. PhD. Thesis, Yale university, Yale, CT, US, 1986.*
5. *Bareiss R., Porter B., Murray, 'Supporting start-to-finish development of knowledge bases', Machine Learning, Vol.4, p. 259-283, 1989.*
6. *Behera B.K., Muttagi S.B., Arun G., Panwar U., 'Expert systems for engineering of technical textiles', Indian Textile Journal, p. 21-23, 2004.*
7. *Biondo S.J., 'Fundamentals of expert systems technology: principles and concepts', Ablex, Norwood, NJ, ISBN-89391-701, 1990.*
8. *Birnbaum L., Collings G., 'Remindings and engineering design themes: a case study in indexing vocabulary'. Proceedings of the Second Workshop on Case-based Reasoning, Pensacola Beach, Florida, 1989.*
9. *Booch G., 'Object-oriented design with applications', Redwood City, California: Benjamin/Cummings, 1991.*
10. *Buchanan B., Smith R., 'Fundamentals of expert systems', Annual Review of Computer Science, Vol. 3, p. 23-58, 1988.*
11. *Chan C.W., Chen L.L., Geng L., 'Knowledge engineering of an intelligent case-based system for help-desk operations', Expert Systems with Applications, Vol. 18, No.2, p. 125-132, 2000.*
12. *Cheung C.F., Lee W.B., Wang W.M., Chu K.F., Chu S., 'A multi-perspective knowledge-based system for customer-service management', Expert Systems with Applications, Vol. 24, No. 4, p. 457-470, 2003.*
13. *Choy K.L., Lee W.B., Lo V, 'Design of case-based intelligent supplier relationship management system – the integration of supplier rating system and product code system', Expert Systems with Applications, Vol. 25, No 1, p. 87-100, 2003.*
14. *Coad P., Yourdon E., 'Object-oriented analysis', Eaglewood Cliffs, New Jersey: Prentice-Hall, 1990.*
15. *Convert R., Schacher L., Pierre V., 'An expert system for the dyeing recipes*

determination', Journal of Intelligent Manufacturing, Vol. 11, p. 145-155, 2000.
16. *Cunningham P., Bonzano A. 'Knowledge engineering issues in developing a case-based reasoning application', Knowledge-based Systems, Vol. 12, No. 7, p. 371-379, 1999.*
17. *David B.S., 'Principles for case presentation in a case-biased aiding system for lesson planning'. Proceedings of the Workshop on Case-based Reasoning, Madison Hotel, Washington, 8-10 May, 1991.*
18. *Doraiswamy I., Basu A., Chellamani K.P., Kumar P.R., 'Fabric engineering using artificial neural networks', Colourage Annual, p. 93-107, 2005.*
19. *Fan J., Newton E., Au R., 'Predicting garment drape with a fuzzy-neural network', Textile Research Journal, Vol. 71, No. 7, p. 605-608, 2001.*
20. *Goodman M., 'CBR in battle planning'. Proceedings of the American Association For Artificial Intelligence, AAAI-86, August 1986, Philadelphia, PA, US, 1989.*
21. *Hinrichs T.R., 'Problem solving in open worlds'. Lawrance Erlbaum Associates, 1992.*
22. *Hunt L., 'Case-based diagnosis and repair of software faults', Expert Systems, Vol. 14, No. 1, p. 15-23, 1997.*
23. *Hussain T., Wardman R.H., Shamey R., 'A knowledge-based expert for dyeing of cotton. Part 1: Design and development', Coloration Technology, Vol. 121, p. 53-58, 2005.*
24. *Johnson R.E., Foote B., 'Designing reusable classes', Journal of Object-Oriented Programming, Vol. 1, No. 2, p. 22-35, 1988.*
25. *Ignizio J.P., 'Introduction to expert systems: the development and implementation of rule-based expert systems', McGraw-Hill, ISBN 0-07-909785-5, 1991.*
26. *Kuo C.J., Lee C., Tsai C., 'Using neural network to identify fabric defects in dynamic cloth inspection', Textile Research Journal, Vol. 73, No. 3, p. 238-244, 2003.*
27. *Lau H.C.W., Jiang B., Lee .B., Lau K.H., 'Development of an intelligent data mining system for a dispersed manufacturing network', Expert Systems Vol. 18, No. 4, p. 175-185, 2001.*
28. *Lau H.C.W., Choy K.L., Lau P.K.H., Tsui W.T., Choy L.C., 'An intelligent logistics support system for enhancing the airfreight forwarding business', Expert Systems, Vol. 25, No. 5, p. 253-268, 2004.*
29. *Lee D., Lee K.H., 'An approach to case-based system for conceptual ship design assistant'. Expert Systems with Applications, Vol. 16, No.2, p. 97-104, 1999.*
30. *Lewandowski S., Stanczyk T., 'Identification and classification of spliced wool combed yarn joints by artificial neural networks. Part 1: Developing an artificial neural network model', Fibres and Textiles in Eastern Europe, Vol. 13, No. 1, p. 39-43, 2005.*
31. *Liao S.H., 'Case-based decision-support system: architecture for simulating military command and control', European Journal of Operational Research, Vol. 123, No. 3, p. 558-567, 2000.*
32. *Lin J., Lin C., Tsai I., 'Applying expert systems and fuzzy logic to an intelligent diagnosis system for fabric inspection', Textile Research Journal, Vol. 65, No. 12, p. 697-709, 1995.*
33. *Linden V., 'Applications of Bayesian decision theory to intelligent tutoring systems', Textile Research Journal, Vol. 54, p. 77-82, 1986.*
34. *Liu Y., Geng Z., 'Three-dimensional garment computer aided intelligent design', Journal of Industrial Textiles, Vol. 33, No. 1, p. 43-54, 2003.*
35. *Micallef J., 'Encapsulation, reusability and extendibility in object-oriented programming languages', Journal of Object-Oriented Programming, Vol. 1, No. 1, p. 12-36, 1988.*
36. *Ng W.W.M, 'Development of an expert system on denim fabric', Proceedings of the World Conference on Asia and World Textiles, p. 609-627, 1993.*
37. *Morpurgo R., Mussi S., 'I-DSS: an intelligent diagnostic support system', Expert Systems, Vol. 18, No. 1, p. 43-58, 2001.*
38. *Nygaard K., 'Concepts in object-oriented programming', ACM SIGPLAN Notices, Vol. 21, No. 10, p. 128-132, October 1986.*
39. *Pascoe G. A., 'Elements of object-oriented programming', Byte Vol. 11, No. 8, p. 139-144, 1986.*
40. *Pham T.T., Chen G., 'Some applications of fuzzy logic in rule-based expert systems', Expert Systems, Vol. 19, No. 4, p. 208-223, 2002.*
41. *Rawal A., Prasad P., Potluri P., Steele C., 'Geometrical modeling of the yarn paths in three-dimensional braided structures', Journal of Industrial Textiles, Vol. 35, No.2, p. 115-135, 2005.*
42. *Rentsch T., 'Object-oriented programming', ACM SIGPLAN Notices, Vol. 17, No. 9, p. 51-57, 1982.*
43. *Robson D., 'Object-oriented software systems', Byte Vol. 6, No. 8, p. 74-86, 1981.*
44. *Srinivasan K., Dastoor P.H., Radhakrishnaia P., Jayaraman S., 'FDAS: a knowledge-based framework for analysis of defects in woven textile structures', Journal of the Textile Institute, Vol. 83, No. 3, p. 431-448, 1992.*
45. *Sycara E.P., 'Resolving adversarial conflicts: an approach to integrating case-based and analytic methods'. Technical report GIT-ICS-87/26, Georgia Institute of Technology, School of Information and Computer Science, Atlanta, GA, 1987.*
46. *Tsai I., Lin C., Lin J., 'Applying artificial neural network to pattern recognition in fabric defects', Textile Research Journal, Vol. 65, No. 3, p. 123-130, 1995.*
47. *Tunstel E., Howard A., Seraji H., 'Rule-based reasoning and neural network perception for safe off-road robot mobility', Expert Systems, Vol. 19, No. 4, p. 191-200, 2002.*
48. *Waterman D. A., A guide to expert systems. Addison Wesley Publishing, Computer, 1986.*
49. *The WIRA textile book, Rae, A., Bruce, R (eds.), Wira, B97-B114, Thornton and Pearson (printers) LTD., Bradford, UK, 1982.*
50. *Xu L.D., Li L. X., 'A hybrid system applied to epidemic screening', Expert Systems, Vol. 17, No. 2, p. 81-89, 2000.*

Received 29.09.2006 Reviewed 18.10.2006

Textile Faculty, TUŁ 1947-2007

Celebration of the 60th anniversary

of the Faculty of Engineering and Marketing of Textiles (formerly Textile Faculty), Technical University of Łódź

8 October 2007

Invitation

Rector Professor Jan Krysiński Ph.D., D.Sc., Dean Prof. Izabella Krucińska Ph.D., D.Sc., and the Faculty Senate have the honour of inviting graduates and friends to a celebration of the 60th anniversary of the Faculty of Engineering and Marketing of Textiles, the Technical University of Łódź (TUŁ), on 8 October 2007.

After the ceremony, the 9th International Conference IMTEX'2007 will be opened, and then the first day's lectures presenting the scientific achievements of the academic staff members will be given.

Honorary committee:

Chairman:
Prof. Jan Krysiński Ph.D., D.Sc., Rector of the TUŁ

Members:
- Prof. Izabella Krucińska Ph.D., D.Sc., Dean of the Faculty of Engineering and Marketing of Textiles, TUŁ
- Prof. Witold Łuczyński Ph.D., Eng., President of the Polish Textile Association
- Julian Bąkowski M.Sc. Eng., President of the Association of the Graduates of the TUŁ

Programming committee:

Chairman:
Prof. Janusz Szosland Ph.D., D.Sc.

Vice-Chairman:
Prof. Jerzy Zajączkowski Ph.D., D.Sc.

Organising committee:

Chairman:
Marek Snycerski Ph.D., D.Sc., Prof. TUŁ

Vice-Chairman:
Bogdan Ignasiak Ph.D., Eng.

For more information, please contact:
Marek Snycerski Ph.D., D.Sc., Prof. TUL
Faculty of Engineering
and Marketing of Textiles
ul. Zeromskiego 116, 90-924 Łódź, Poland
e-mail: Marek.Snycerski@p.lodz.pl