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

Integrating an Object-oriented Approach and Rule-based Reasoning in the Design of a Fabric Fault Advisory Expert System

Abstract
This paper describes the design of an expert system that not only offers advice to both textile and non-textile users on fabric faults, but also assists in the diagnosis of fabric faults. The expert system integrates both object-oriented and rule-based reasoning approaches. The system is based on the premise that the design of a component should be separated from the implementation details. In short, this means that the rules that manipulate a set of objects exist separately from the same set of objects. Separation of object definitions from rules that manipulate them in the knowledge base, and separation of the object-based knowledge base from the object-based inference engine that infers conditions, means that any changes made to any one of these components do not necessarily mean that changes to the other components have to be made, and also that each of these components is autonomous. Dependencies between components in an object in the object-oriented approach, the extendibility of such systems, the inheritance properties, and reusability of objects are just some of the advantages of adopting an object-oriented approach.

Key words: *object-oriented, rule-based, expert system, fabric faults.*

Introduction

Expert systems can be of great use to both textile and non-textile users. They can offer substantial support when advice is sought on textile issues, with fabric fault analysis being one of these. When expert systems are used in the textile sector for fabric fault analysis, they usually not only contain basic knowledge on fabric faults, but also rules regarding evidence and procedures leading to the diagnosis of faults. A clearer insight into the diagnosis patterns is presented.

The aim of this paper is to describe an expert system which can identify common fabric faults and can also act as a source of information on fabric faults. The system can be interrogated to give an analysis of the characteristics of fabric faults, and solutions for rectifying these faults.

The significance of the expert system is:
- to enable users to query the knowledge base for fabric fault identification on the basis of a given set of characteristics;
- to provide fabric manufacturers with information on solutions to rectify identified fabric faults;
- to act as a training guide for workers and technicians in fabric manufacture;
- to provide fabric manufacturers with technical information.

The design of the system is based on an object-oriented approach coupled with rule-based reasoning. On the other hand, separation of object definitions from rules that manipulate them in the knowledge base, and separation of the object-based knowledge base from the object-based inference engine, are supported in the system. This simply means that any changes made to any one of these components do not necessarily mean that changes to the other components have to be made, as they are only loosely coupled. Extending the system by adding new objects is supported.

Object-oriented approach

Objects are autonomous entities that have a state and respond to messages. An object is an entity that encapsulates some private state information or data, a set of associated operations or procedures that manipulate the data, a possible thread of control so that collectively they can be treated as a single unit. A system is object-based if it supports objects and object-oriented if it supports the concept of inheritance. Inheritance is a mechanism that permits objects to be developed from existing objects simply by specifying how the new objects differ from the originals. This means that an object may inherit the behaviour and operations of a super object. Object-oriented paradigms have been discussed in various publications [3, 4, 8].

According to [37], object-oriented systems are uniform in that all items are objects and no object properties are visible to an outside observer. All objects communicate using the mechanism of message passing, and the processing activity takes

place inside objects. Inheritance allows classification, sub-classification and super-classification of objects which permits their properties to be shared. In [36], the object-oriented approach is defined in terms of encapsulation, data abstraction, methods, messages, inheritance and dynamic binding for object-oriented languages. [38, 42] emphasise the idea of message-passing between objects and dynamic binding as fundamental to the object-oriented approach.

Abstraction, in terms of object-oriented concepts, is a technique that involves a 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 [6]. Forming an abstraction in terms of objects is one of the fundamental tenets of the object-oriented paradigm. In the application of different object-oriented methods, [26] has shown that different methods facilitate the abstraction of different objects within the same problem situation. Data-driven methods, such as object-oriented analysis [8] and the object-modelling technique [40], might identify and select objects by concentrating on the data qualifying a 'thing' or concept in the situation. Alternatively, process-driven methods, such as that of [45], may identify and select objects by focusing on the processes in the problem situation.

Reusing software components which are already available 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 reusable components exists, browsing components to identify opportunities for reuse should take precedence over writing new ones from scratch. Inheritance is an object-oriented technique that boosts reusability [7, 17, 23, 31].

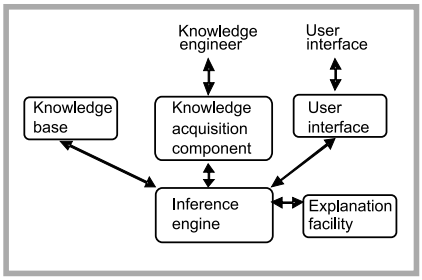

Figure 1. *General structure of an expert system.*

Components of an expert system

The components of an 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, which contains the information that the expert system uses to make decisions. This information presents expertise gained from top experts in the field, in the form of facts and rules. Facts are minimal elements of the knowledge which must be identified before anything else. For example, 'A bar is a fabric 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 picking space is different from the normal cloth, then a pick bar results'. A frame is another approach to capture and store knowledge in a knowledge base. It relates an object to various facts or values. Expert systems using frames to store knowledge are also called frame-based expert systems. Semantic nets, neural networks and fuzzy logic are other methods of representing knowledge in expert systems.

Inferencing. The purpose of the inference engine is to seek information and form relationships from the knowledge base and provide answers. It determines which rules will be applied to a given question, and in what order, by using information in the knowledge base. The inference engine drives the system by drawing an inference from relating user-supplied facts to a knowledge-base rule, and then proceeding to the next fact and rule combination [5].

Two types of inference methods which are typically implemented in expert systems are backward and forward chaining. Backward chaining is an approach that starts with the goal, e.g., 'Which fabric fault is it?' and works through a potential thesis until it reaches a fact that supports the thesis. A forward-chaining inference engine is goal-oriented in the sense that it tries to prove a goal or rule conclusion by confirming the truth of all its premises. These premises may themselves be conclusions of other rules. It is a method that begins with a set of known fact or attributes values and applies these values to rules that use them in their premise.

Knowledge acquisition. Most expert systems continue to evolve over time. New rules can be added to the knowledge base by using the knowledge acquisition subsystem.

Explanation subsystem. 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 carried out in a sub-system known as the explanation subsystem. This enables the subsystem 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.

Related work

Software developers would not want to give up the capabilities of modern-day development environments and libraries that accompany the object-oriented approach. So ideally, the two paradigms of object-oriented approach and rule-based reasoning are both available [21, 32]. The systems that combine the two paradigms of object-orientation and rule-based reasoning implement a rule-based system as a library in object-oriented programming languages. Examples of these have been given in [15, 24 and 34]. This research takes an approach of designing the whole system using objects and applying rules in object form to manipulate these objects.

In textiles, rule-based expert systems have been applied in dyeing recipe determination [9, 10, 20], for the selection of fluorescent whiteners [1], for three dimensional computer-aided intelligent design of garments [29], for fabric engineering [2, 33] and the analysis of defects in textiles [41].

Expert systems may incorporate other knowledge representation methods, such as frames, semantic nets, neural networks and fuzzy logic, besides rule-based reasoning. In textiles, neural networks have been applied in the identification of fabric defects [25, 43], in prediction of garment drape [16] and in fabric engineering [14]. Fuzzy logic has been applied to an intelligent diagnosis system for fabric inspection [27], while Bayesian networks have been applied in intelligent tutoring system in textiles [28].

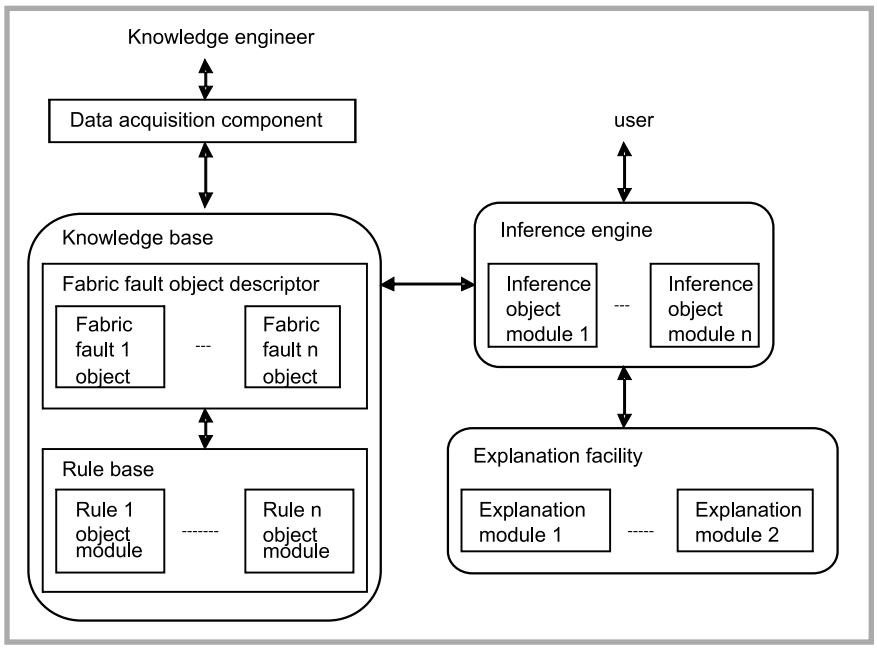

Figure 2. *Components of the proposed architecture.*

Several approaches to developing software applications with rule-based knowledge advocate making the knowledge explicitly separating it from other functionality have been looked at in [11, 13, 18, 39]. Integration of rule-based knowledge and object-oriented functionality with linguistic symbiosis is examined in [12].

Components of the system's architecture

The proposed architecture identifies objects for the knowledge base, the inference engine and explanation facility, as shown in Figure 2. Since the knowledge base is made up of both facts and rules, the facts are represented as objects that describe the fabric faults in the form of their characteristics and solutions. The rules are represented as operations to store and query these characteristics and solutions. The objects in the knowledge base, inference engine and explanation are stored as objects in the form of software modules. For such objects the standard inheritance mechanisms of object-oriented approach are applied. The system is therefore modular.

Software process definitions have to be changed over time. An architecture that separates the rule base and the object descriptor in the knowledge base, the knowledge base and inference engine, gives that flexibility by allowing changes to each of the components without affecting the other components [13].

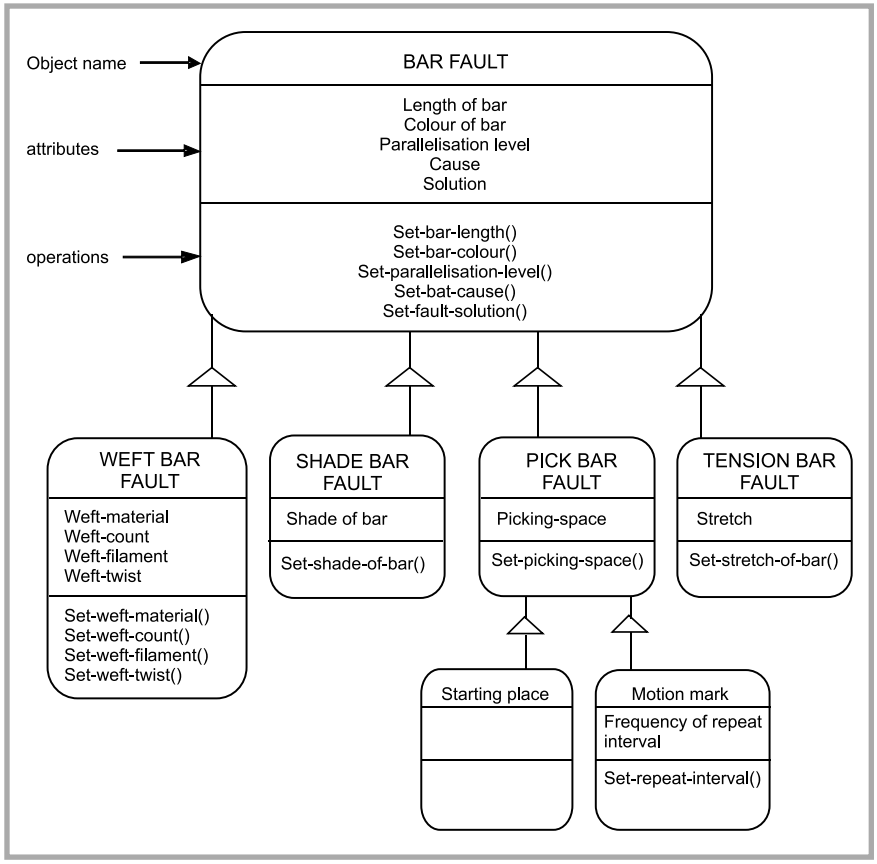

Figure 3. *Object diagram.*

Fabric faults

This section describes a limited number of fabric faults, their origins and solutions as drawn from [44] for the purposes of this paper. This information is used in the design of the expert system.

BAR. This is a general term covering a number of specific faults in the form of a bar, running across the full width of a piece, which differs in appearance from the adjacent normal cloth. It may be shady or solid in appearance, and may or may not run parallel with the picks.

- WEFT BAR: This is a bar that is solid in appearance, is clearly defined, runs parallel to the picks and contains a weft (or wefts) that is different in material, count, filament, twist, lustre, colour or shade from the adjacent normal weft.
- SHADE BAR: This is a bar that has developed a different shade from the adjacent cloth during or subsequent to dyeing and finishing, owing to damage or contamination of otherwise normal cloth or weft yarn prior to weaving.
- PICK BAR (STARTING PLACE): This is a bar in which the picking space is different from that in the normal cloth. In this case, there is an isolated narrow bar running parallel with the picks, starting abruptly and gradually shading away to normal cloth. This is due to an abrupt change in pick spacing followed by a gradual return to normal pick spacing. Such a bar may occur on restarting weaving after (i) finding a pick, (ii) unweaving or pulling back, (iii) prolonged loom stoppage.

- PICK BAR (MOTION MARK or WEAVING BAR): This is a bar in which the picking space is different from that in the normal cloth. In this case, the bar usually shades away to normal cloth at both its edges. It owes its appearance to a change in pick spacing, and may repeat at regular intervals throughout an appreciable length or even the whole length of a piece. Such a bar is the result of some mechanical fault on the loom, such as faulty gearing in the take-up motion, bent beam gudgeons, uneven or eccentric beam ruffles, uneven bearing surfaces at some point in the let-off motion, etc.
- TENSION BAR: This is a bar composed of weft yarn that has been stretched more or less than the normal weft, prior to or during normal weaving. This abnormal stretch may have been imposed during winding, by faulty manipulation or by some mechanical fault in the machine; during weaving by incorrect tensioning in the shuttle, or may have arisen owing to the faulty yarn having been excessively moistened at some stage, and consequently stretched more than the normal yarn under the normal applied tensions.

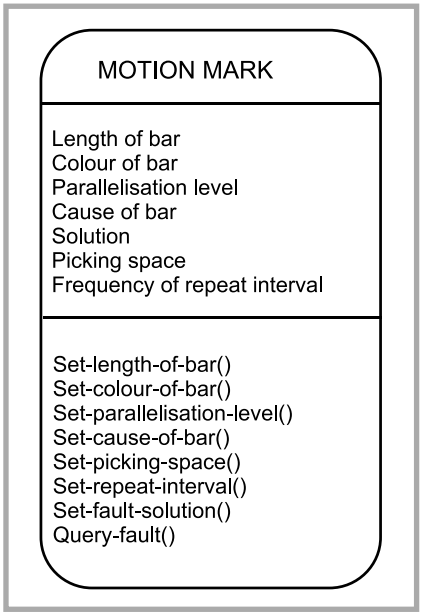

Figure 4. *Object structure for the motion mark.*

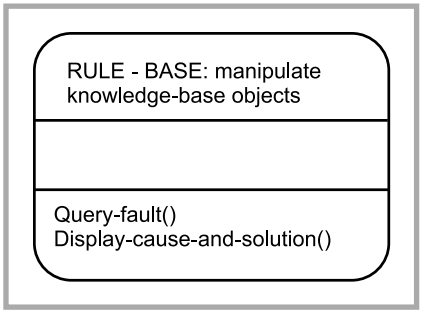

Figure 5. *Rule-base object.*

The knowledge base

The various fabric faults are represented as objects in the knowledge base. An object consists of the characteristics of the fabric fault, and the methods that manipulate these characteristics. For example, the characteristics of a bar include its length, colour, cause, solution and level of parallelisation to the flow of the fabric.

Figure 3 shows the structure of the object descriptor in the knowledge base. Each fabric fault shows its attributes (e.g. length, colour, cause and solution to a bar) and the operations to accept the characteristics.

Inheritance is one of the characteristics of objects. The pick bar (starting place) and pick bar (motion marks) objects are both variations of the pick bar. Therefore they will inherit attributes of the pick bar object such as the picking space in addition to their characteristics. That means, therefore, that a pick bar (staring place) will inherit the characteristics of both the super objects, pick bar and bar, in addition to its own, since the pick bar also inherits from the object bar. Figure 4 shows the virtual structure of the object pick bar (motion mark) after it has inherited the attributes and methods of the super objects.

The rule base

The idea of rule-based systems is to represent a domain expert's knowledge in a form called rules. 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 rules query-fault() and display-cause-and-solution() would be in a separate module (as shown in Figure 5) in the architecture, as they manipulate the objects in the descriptor.

To reduce the complexity of coming up with rules, a decision tree as shown in Figure 6 is designed for the problem.

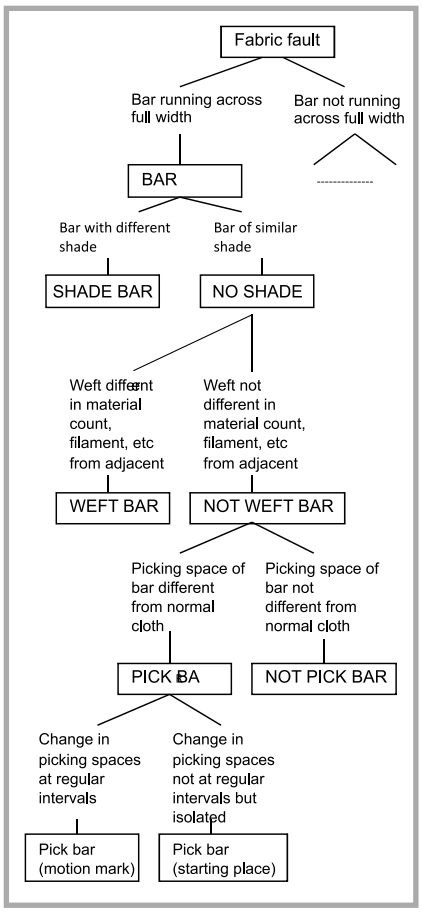

Figure 6. *Decision tree for fabric faults.*

The resultant rules are as follows:

RULE WEFT BAR:
IF the bar is running across full width
AND weft different in material count, filament, etc, from adjacent weft
THEN weft bar process

RULE SHADE BAR:
IF bar is running across full width
AND bar a different shade from material
THEN shade bar process

RULE PICK BAR:
IF the bar is running across full width
AND picking space of bar different from normal cloth
AND running parallel with picks
THEN pick bar process

RULE MOTION MARK:
IF the bar is running across full width
AND picking space of bar different from normal cloth
AND change in picking spaces repeats at regular intervals
THEN motion mark process

RULE TENSION BAR:
IF bar is running across full width
AND weft yarn stretched more or less than normal weft
THEN tension bar process

The processes identified such as pick bar process, motion mark process, etc, will manipulate the objects in the knowledge base descriptor.

Inference engine

The contents of the inference engine are as follows:

```
method fabric faults
{ if (bar running across full width)
 call method BAR
 else
 call method -----}

method bar
{ if (bar with different shade)
 fault name is SHADE BAR
 else
 if bar of similar shade
call method NO SHADE}

method no shade
{ if (weft different in any of material, count, filament, etc, from adjacent)
 fault name is WEFT BAR
 else
 call method NOT WEFT BAR}

method not weft bar
{  if (picking space of bar different from normal cloth)
 call method PICK BAR}

method pick bar
{ if (change in picking spaces at regular intervals)
 call method PICK BAR(motion mark)
 else
 call method PICK BAR (starting place)}
```

The user interface

The user interface accepts the characteristics of faults identified, and comes up with a diagnosis of the fault, cause and solution. Characteristics such as length, colour, shade, material of bar, count, twist, stretch and picking space are captured, whichever is available.

Advantages of approach

The object-oriented approach has the following advantages: when the expert system is large, complexity is reduced through modularisations, that is, by subdividing the system into manageable size components, such as objects, and establishing well-defined relationships between them. The internal design of each object is localised so that it does not depend on the internal design of another component. The design concepts are separated from the implementation details. That means that rules are developed separately from the objects that they manipulate. Objects can be reused. They are written and debugged once, and then matched to form new applications.

The advantage of separation of the various components is that each of these is autonomous. What should be well-managed are the relationships between them. Unlike the other systems that implement a rule-based system as a library in object-oriented language, this architecture extends this further by applying object technologies to every single component of the expert system, including the rule base.

Conclusion

In this paper, an expert system architecture that marries object-technology and rule-based reasoning is applied to fabric fault analysis. The architecture supports separation of object definitions from rules that manipulate them in the knowledge base, and separation of the object-based knowledge base from the object-based inference engine.

References:

1. *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.*
2. *Behera, B.K., Muttagi, S.B., Arun, G., Panwar, U., The Indian Textile Journal, p. 21-23, 2004.*
3. *Booch, G., 'Object-oriented design with application', Freeman, F.P., Wasserman, A.I. (eds), Benjamin/Cummings, Redwood City, California, 1991.*
4. *Booch, G., Tutorial on software design techniques, IEEE press, p. 420-436, 1983.*
5. *Buchanan, B.G., Shortcliffe, E.H., 'Rule-based expert systems: the MYCIN experiment of the Stanford heuristic programming project', Addison Wesley, Reading, 1985.*
6. *Capretz, L.F., 'A brief of the object-oriented approach', ACM SIGSOFT, Software Engineering Notes, Vol. 28, No. 2, p. 1, 2003.*
7. *Capretz, L.F., Lee, P.A., 'Reusability and life cycle issues within an object-oriented design methodology', Ege, R., Singh, M., Meyer, B. (eds). Proceedings of Technology of Object-oriented Languages and Systems (TOOLS USA'92), Prentice Hall, Englewood Cliffs, New Jersey, p. 139-150, 1992.*
8. *Coad, P., Yourdon, E., Object-oriented analysis, Prentice Hall, Englewood Cliffs, New Jersey, 1991.*
9. *Convert, R., Schacher, L., Pierre,V., 'BATEM: an expert system for the dyeing recipes determination', Proceedings of the World Conference of the Textile Institute, Vol. 2, p. 275-276, 1997.*
10. *Convert, R., Schacher, L., Pierre, V., 'An expert system for the dyeing recipes determination', Journal of Intelligent Manufacturing, Vol. 11, p. 145-155, 2000.*
11. *Date, G., What, not how: the business rules approach to application development, Addison Wesley, 2000.*
12. *D'Hondt, M., Gybels, K., Jonckers, V., 'Seamless integration of rule-based knowledge and object-oriented functionality with linguistic symbiosis', (SAC'04), p. 1328-1335, 2004.*
13. *Dlodlo, N., Bamford, C., 'Separating application functionality from the user interface', Proceedings of EUROMICRO'97, Sept 1-4, 1997.*
14. *Doraiswamy, I., Basu, A., Chellamani, K.P., Kumar, P.R., 'Fabric engineering using artificial neural networks, Colourage Annual, p. 93-107, 2005.*
15. *Eclipse. Website of the Haley Enterprise Inc. http://www.haley.com.*
16. *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.*
17. *Gossain, S., Anderson, B., 'An iterative design model for reusable object-oriented software', ACM SIGPLAN Notices, Vol. 25, No. 10, p. 12-27, 1990.*
18. *Halle, von, B., Business rules applied, Wiley Publishers, 2001.*
19. *Henderson-Sellers, B., Constantine, L.L., 'Object-oriented development and functional decomposition', Journal of Object-oriented Programming, Vol. 3, No. 5, p. 11-17, 1991.*
20. *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.*
21. *Jackson, P., Introduction to expert systems, Addison-Wesley, 1986.*
22. *Jacobson, I., 'Object-oriented development in an industrial environment', ACM SIGPLAN Notices, Vol. 22, No. 12, p. 183-191, 1987.*
23. *Johnson, R.E., Foote, B., 'Designing reusable classes', Journal of Object-Oriented Programming, Vol. 1, No. 2, p. 22-35, 1988.*
24. *JRules 4.0, Technical white paper from ILOG.*
25. *Kuo, C.J., Lee, C., Tsai, C., 'Using neural networks to identify fabric defects in dynamic cloth inspection', Textile Research Journal, Vol. 73, No. 3, p. 238-244, 2003.*
26. *Liang, Y., Newton, M.A., Robinson, H.M., 'The use of object models for information systems analysis', Proceedings of the Fourth International Conference on Information Systems Development, Bled, Slovenia, p. 625-634, 1994.*
27. *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.*

28. *Linden, V., 'Applications of Bayesian decision theory to intelligent tutoring systems,' Textile Research Journal, Vol. 54, p. 77-82, 1986.*
29. *Liu, Y., Geng, Z., 'Three-dimensional garment computer aided intelligent design', Journal of Industrial Textiles, Vol. 33, No. 1, p. 43-54, 2003.*
30. *Lorenson, D., 'Object-oriented design', CRD Software Engineering Guidelines, Generic Electronic Company, 1986.*
31. *Micallef, J., 'Encapsulation, reusability and extendibility in object-oriented programming languages', Journal of Object-Oriented Programming, Vol. 1, No. 1, p. 12-36, 1988.*
32. *Milliken, K.R., Finkel, A.J., Klein, D.A., Waite, N.B., 'Adding rule-based techniques to procedural languages'. Proceedings of the First International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, p. 185-194, 1988.*
33. *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.*
34. *OPSJ4.1, Manual by Charles L. Forgy from Production Systems Technologies Inc.*
35. *Pachet, F., 'On the embeddability of production rules in object-oriented languages', Journal of Object-Oriented Programming, Vol. 8, No. 4, p. 19-24, 1995.*
36. *Pascoe, G.A., Elements of object-oriented programming, Byte, Vol. 11, No. 8, p. 139-144, 1986.*
37. *Rentsch, T., 'Object-oriented programming', ACM SIGPLAN Notices, Vol. 17, No. 9, p. 51-57, 1982.*
38. *Robson, D., 'Object-oriented software systems', Byte, Vol. 6, No. 8, p. 74-86, 1981.*
39. *Ross, R.G., Principles of the business rule approach, Addison Wesley, 2003.*
40. *Rumbaugh J., Blaha, M., Premerlani,W., Eddy, F., Lorrenson, W., 'Object-oriented modelling and design', Prentice Hall, Englewood Cliffs, New Jersey, 1991.*
41. *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.*
42. *Thomas, D., 'What's in an object', Byte, Vol. 14, No. 3, p. 231-240, 1989.*
43. *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.*
44. *The WIRA textile book, Rae, A., Bruce, R (eds), Wira, B97-B114, Thornton and Pearson (printers) Ltd., Bradford, UK, 1982.*
45. *Wirfs-Brock, R., Wilkerson, B., Wiener, L., 'Designing object-oriented software', Prentice Hall, Englewood Cliffs, New Jersey, 1990.*

Received 29.09.2006 Reviewed 16.04.2007