Processing Technologies, AMPT2003 Vol. 1 (pp. 619-622). Dublin City University, Dublin, Ireland, 8-11 July 2003.

FAILURE MODES AND EFFECTS ANALYSIS THROUGH KNOWLEDGE MODELLING

P.C.Teoh and K.Case
Mechanical and Manufacturing Engineering, Loughborough University, UK

ABSTRACT

Failure Mode and Effect Analysis (FMEA) is a widely used quality improvement and risk assessment tool in manufacturing. Design and process failures recorded through FMEA provides valuable knowledge for future product and process design. However, the way the knowledge is captured poses considerable difficulties for reuse. This research aims to contribute to the reuse of FMEA knowledge through a knowledge modelling approach. FMEA activities are shifted to the conceptual design stage to avoid costly and difficult design changes at later stages of the design process. An object-oriented approach has been used to create an FMEA model. Functional diagrams have been used for the conceptual model. The FMEA model uses functional reasoning techniques to enable automatic FMEA generation from historical data. The reasoning technique also provides a means for the creation of new knowledge. The automatic generation replaces the traditional brainstorming process for FMEA report creation. The sources of the historical data can be from the previous FMEA, failure reports or from the individual designers.

Keywords: FMEA, conceptual design, knowledge representation, causal reasoning

1 INTRODUCTION

FMEA is a technique that identifies the potential failure modes of a product or a process, the effects of the failures, and assesses the criticality of these effects. It provides basic information for reliability prediction, and product and process design [1]. “FMEA is a method of reliability analysis intended to identify failures, which have consequences affecting the functioning of a system within the limits of a given application, thus enabling priorities for action to be set.” [2]. FMEA can be classified into two main types of design FMEA to deal with design activities, such as product design, machine or tooling design and process FMEA to solve problems due to manufacturing processes. FMEA is carried out by a cross-functional team of experts from various departments. The team analyses each component and subsystem of the product for the failure modes and determines the potential causes and effects. The risk of each failure is prioritised based on the risk priority number (RPN). RPN is a decision factor based on the product of three ratings: occurrence, severity and detection. Any improvement plan would be based on the indications from the RPN with the current controls (i.e. the solutions) being implemented for products/processes with high RPNs.

2 PROBLEM DEFINITION

Traditionally, FMEA is used in hard copy or spreadsheet format to capture the potential problems. Although the knowledge is aimed for reuse, as the knowledge grows, it is harder to reuse. A highly manual FMEA system is found to be not user friendly, hard to understand and not very flexible. There is also duplication of information in other documents in the factory. As a result, many companies use FMEA merely to satisfy contractual requirements [3] and users may find FMEA a “tedious and time-consuming activity” [4]. FMEA is often carried late in the design cycle after the design prototype has been built [4]. The changes made at later stages will be very costly.

Much research has been carried out in FMEA for example in the electrical design of automobile systems [4] and mechanical design [5]. Little of this work has reached practical implementation and most mechanical, electromechanical and manufacturing process designs still rely on conventional methods.

The research described here looks at a knowledge modelling approach to overcome the limitations discussed above.

3 FMEA IN CONCEPTUAL DESIGN

According to Pahl and Beitz [6], the design process consists of four phases: design specification, conceptual design, embodiment design and detail design.

Traditional FMEA is suitable for use in the detail design phase where the design solutions have been firmed up and the information required is easier to obtain. To move the FMEA involvement further upstream to conceptual design can be a challenge. Traditional FMEA will not be able to cope with frequent design changes, and conceptual design deals with more abstract and imprecise information. However, conceptual design is very important as a poor design concept can never be compensated at later design stage [7]. Hence, this research attempts to implement FMEA within conceptual design and this requires autiomation of the FMEA generation process. Design changes must be reflected automatically in the FMEA without extra effort during conceptual design.

4 FMEA MODEL

According to Hubka and Eder [8], a transformation system is defined as “a sum of all elements and influences (and the relationships among them and their environment) that participate in a transformation”. Briefly, the elements of a transformation system consist of an operand, a technical process, technical system, human system and active environment. Figure 1 shows the transformation system for a conveyor system

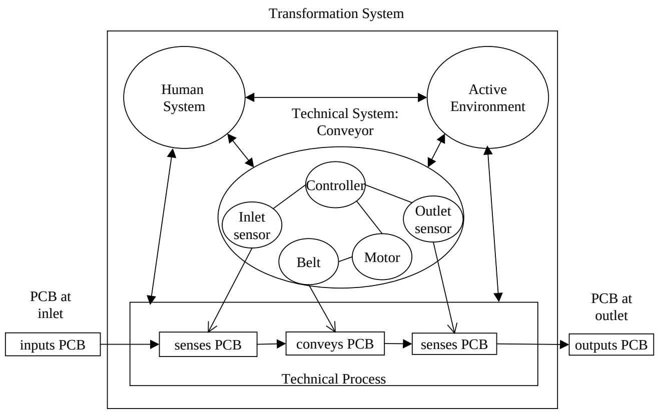

Figure 1 Transformation System for Conveyor

The transformation system can be used to organise the entities in a component library. To completely define the transformation system, entities are organised in five main classes: Technical System, Technical Process, Human System, Active Environment and Operand.

A Technical System represents the class for design artifacts and equipment used for manufacturing processes. It is further categorised into two main classes, Machine and Product. The Machine represents machinery involved in the manufacturing process, such as the conveyor. The Product deals with entities from the design artifacts. A machine or a product can be made up of sub-assemblies and components. The Machine Component is used to represent the sub-assemblies and components for a Machine, whereas the Product Component is used for sub-assemblies and components of a Product.

The decision on whether an entity is included as a machine or a product is purely based on the users’ point of view. A conveyor in a PCB assembly plant will be in the machine class, whereas for conveyor manufacturer it would be in the product class.

Technical Process is a class for manufacturing processes and contains a series of Function Units which define the sub-processes and process steps to accomplish a manufacturing objective. For example, radio assembly process is included in this class.

Human System is a class for human objects, such as users, machine operators and customers who are involved in handling machinery, products or executing manufacturing processes.

The Active Environment includes environmental elements which affect the other categories, and can be physical phenomena such as heat, water or interference.

Operand is the class for entities that are acted upon by others. They are usually involved directly in the end results of products and processes. The product or process functions are achieved by state changes to the operands. An object diagram for the transformation and components for the conveyor system is as shown in Figure 2.

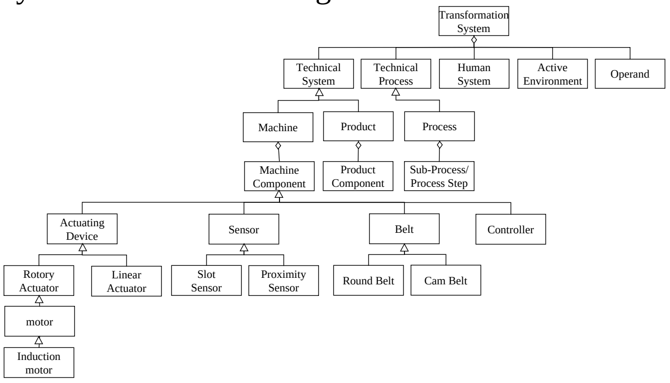

Figure 2. Transformation

Entity is defined as the super class for the elements in the transformation system. It is used as the basic class of all the physical objects. An Entity is characterised by its properties and states. Property is a class to represent the properties that define the Entity. In turn, State provides the value of a Property which defines the condition of the Entity. For example, a sensor can have a property “functionality” and state “not functioning”.

Generic Functions are used to represent the generic grouping of all functions involved in designs and processes and have been developed based on the functional basis developed by Hirtz et al [9]. Figure 3 shows an example for a Generic Function tree. A Generic Function can have many associated Behaviours. The Behaviour in this context is considered as a description to define the status of the Generic Function. For example, a generic function "conveys" may have behaviours "conveying" and "not conveying".

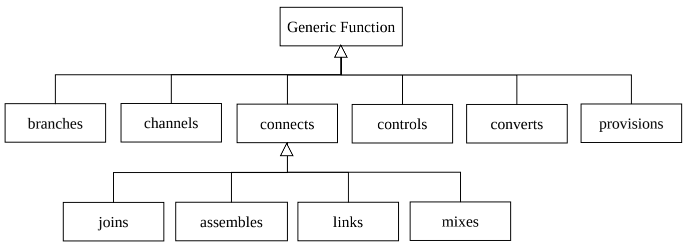

Figure 3. Generic Function Tree (after [9])

A conceptual model can be used to represent the interactions among the entities in the transformation system. The conceptual model can be illustrated by the following examples.

In a conveyor system, a motor is used to move a belt. The motor interacts with the belt through a function called "conveys". The motor is an entity known as the operator which acts on another entity, the belt. "Conveys" is a generic function term to represent the "move" action. The belt, which is the receiver of the action "conveys", is also known as the operand of the relationship. The relationship among an Operator, an Operand and a Generic Function is known as a Function Unit. Operator and Operand are instances of Entity. An Operator in one instance can become an Operand in another. For example, a belt is an operand in the function unit, "motor conveys belt", but it can be an operator for a new function unit, "belt conveys PCB".

The Functional Units for the conveyor system can form a functional diagram which is the graphical representation of the conceptual model (figure 4).

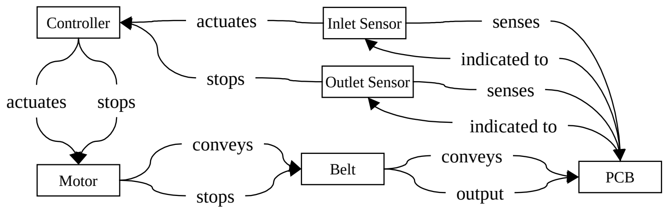

Figure 4. Functional Diagram

A cause and effect propagation method can be used to simulate the actual behaviour of a design in the real world. In a functional model, a state change in one entity will affect the status of the inter-related entities. The propagation is carried out through the Behaviour of a Generic Function. The relationship between an entity State and a function Behaviour is characterised by preconditions and postconditions.

The state of the operator will determine the Behaviour of the Generic Function within a Function Unit. This is the precondition relationship. The Behaviour will in turn decide the state of the operand within the Function Unit. This is termed the postcondition relationship. For example, an inlet sensor senses a PCB. The operator is "inlet sensor", the generic function is "senses" and the operand is "PCB". If a state description "sensor failure" is introduced to the operator "inlet sensor", the behaviour for the function "senses" is "not sensing", and the operand state is "PCB not sensed". The precondition relationship is "sensor failure – not sensing", and the postcondition relationship is "not sensing – PCB not sensed".

In a functional model, the interaction between Function Units is carried out through the Entity itself as, in most cases, an operand of a Function Unit is an operator of the next Function Unit. Hence, if the state change occurs, the changes will be propagated and the series of preconditions and postconditions create a causal chain for a particular state change event.

In order to facilitate cause and effect propagation, a functional diagram must be able to respond to stimulation or changes of state in its components. This response is driven by the causal reasoning. The causal reasoning knowledge is stored in Precondition and Postcondition in the forms of "operator failure state – failure behaviour" and "failure behaviour – operand failure state".

The causal reasoning is based on two basic assumptions that (a) there exists a state of an operator where if there is a change to that state, it will cause its functional behaviour to change accordingly and (b) there exists a functional behaviour where if there is a change to that behaviour, it will cause the corresponding operand to change its state accordingly.

The Precondition and Postcondition gain knowledge through historical data extracted from failure reports and the FMEA. For a particular function unit, the operator state and the behaviour of a failure event form a set of preconditions. The behaviour and the state of the operand form the postcondition of the same event. Hence, with the accumulated events being recorded, precondition and postcondition tables will be formed. Using this approach, the static knowledge is confined to the entities and their functions, but not to the function units. During the reasoning process, it is possible to create new knowledge by matching the precondition and postcondition knowledge with similar failure behaviour. Using the same conveyor example, the function of the motor is to move the conveyor belt. The belt in turn is intended to move the PCB laid on top of the belt. At an event when the motor fails due to a burnt fuse, the belt will not move, and neither does the PCB. Hence the knowledge fragment captured in precondition and postcondition tables can be arranged as follows:

Precondition Table

Operator	Generic Func.	Precondition
Motor	conveys	fuse burnt – not conveying
Belt	conveys	belt not moving – not conveying

Postcondition Table

Generic Func.	Operand	Postcondition
conveys	belt	not conveying – belt not moving
conveys	PCB	not conveying – PCB not moving

The precondition table defines the behaviour of the motor with the blown fuse, and the behaviour of the belt when it is not moving. The postcondition table provides the knowledge about the response of the belt when it receives the behaviour "not conveying". The postcondition table also provides the knowledge about the response of the PCB when it receives the behaviour "not conveying". This approach provides a modularity for the creation of new knowledge.

In creating a new function unit, the operator, operand and the generic function can be used as keys to find matching states and behaviours in the precondition and postcondition tables. Hence, an entity is able respond to the system through its distinctive "memory". When another designer is creating a design with the new function unit: "motor conveys PCB", the system will search for the operator with the name "motor" with function "conveys" and retrieve the likely precondition (fuse burnt – not conveying). The same process is carried out on the operand "PCB" and function "conveys", and the likely postcondition (not conveying – PCB not moving) is retrieved. Combining this information will result in a new case: "fuse burnt – PCB not moving". Hence, PCB has the knowledge to respond to the motor failure even though the case has never previously existed.

5 CONCLUSION

The current issues faced by FMEA users and the need to migrate FMEA activities to the earlier phase of the design process have prompted this research. This paper has provided the knowledge representation required to build an FMEA model. The model and the proposed reasoning technique form a framework for automatic FMEA generation.

The FMEA model is created based on the transformation system. Component libraries are created and generalised using the object-oriented approach. The concepts of function unit and functional diagram are introduced. The generic function is used as an abstract object to represent functions of designs and processes. Cause and effect propagation is used in the functional diagram with the aid of precondition and postcondition relationships, leading to the idea of new knowledge generation based on two basic assumptions.

ACKNOWLEDGEMENTS

The authors would like to thank Loughborough University and Motorola Technology Malaysia PLC for supporting the research.

REFERENCES

[1] Teng Sheng-Hsien, Ho Shin-Yann, Failure Mode and Effects Analysis – An Integrated Approach for Product Design and Process Control. *International Journal of Quality & Reliability Management*, Vol. 13 No. 5 (1996), pp. 8-26. MCB University Press.

[2] *BS 5760* Part 5, Reliability of systems, equipment and components. Guide to failure modes, effects and criticality analysis (FMEA and FMECA), (1991).

[3] B.G.Dale, P.Shaw, Failure Mode and Effects Analysis in the U.K. Motor Industry: A state-of-the-art Study, *Quality & Reliability International*, (1996).

[4] C.J.Price, D.R.Pugh, M.S.Wilson, N.Snooke, Flame system: automating electrical failure mode & effects analysis (FMEA), *Proc. Annual Reliability and Maintainability Symposium*, (1995), p 90-95.

[5] N.Hughes, E.Chou, C.J.Price, M.Lee, Automating mechanical FMEA using functional models. *Proceedings of the Twelfth International Florida AI Research Society Conference. AAAI Press*, (1999), pp. 394-8.

[6] G.Pahl, W.Beitz, *Engineering Design A systematic Approach*, Second Edition, Springer-Verlag, London, (1996).

[7] W.Hsu, I.M.Woon. Current research in the conceptual design of mechanical products, Singapore, *Computer Aided Design*, Vol 30, No 5, Apr (1998), pp 377-389.

[8] V.Hubka, W.E.Eder, *Theory of Technical Systems: A Total Concept Theory for Engineering Design*, Springer-Verlag Berlin, (1988).

[9] J.Hirtz, R.B.Stone, D.A.McAdams, S.Szykman, K.L.Wood. A Functional Basis for Engineering Design: Reconciling and Evolving Previous Efforts. *Research in Engineering Design* 13(2):65-82, (2001). http://web.umr.edu/~rstone/research/journals/Reconciled_FB-RED.pdf. (2002).