Int. J. Computer Integrated Manufacturing, Vol. 18, No. 4, June 2005, 279–293

Taylor & Francis
Taylor & Francis Group

An evaluation of failure modes and effects analysis generation method for conceptual design

P. C. TEOH* and KEITH CASE

Mechanical and Manufacturing Engineering, Loughborough University, Loughborough, Leics, LE11 3TU, UK.

Failure modes and effects analysis (FMEA) is used in the manufacturing industry to improve product quality and productivity. However, the traditional approach has many shortcomings that affect its effectiveness and limit its usefulness, especially in the early stages of design. Automating the FMEA report generation process seems to answer some of these problems, and there has been much past and on-going research in this area. However, most of the work is limited to specific applications. This paper proposes a method for FMEA generation for a generic application using minimum information during the conceptual design stage. Prototype software has been created for the proposed method. It has been evaluated using case studies from the design and manufacture of two-way radios. The evaluation revealed the feasibility of the proposal, as well as some weaknesses that need further improvement. Generally, the capability of the method to generate FMEA report with minimum information is demonstrated.

1. Introduction

Failure modes and effects analysis (FMEA) was originally used by the US military to evaluate the impact of system and equipment failures on mission success, and the safety of personnel or equipment. Eventually, the manufacturing industry adopted the method for quality improvement and risk assessment in design and manufacture. In 1990, the International Organization for Standardization (ISO) recommended the use of FMEA for design review in the ISO 9000 series (Chen 1996).

FMEA is an analysis tool that identifies potential failure modes of a design or a manufacturing process, and the resultant effects on product quality and functionality. According to BS 5760 Part 5 (1991 p.3), '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.' Generally, there are two types of FMEA: design FMEA and process FMEA.

Design FMEA is used to identify design failures for products, machine or tooling, while process FMEA is applied to manufacturing process analysis. In both cases the effects of the failures are identified and the risks assessed accordingly.

An FMEA team consists of cross-functional members from various departments, including design, production, purchasing, quality assurance, and sometimes may also involve legal personnel. The aim is to ensure that various aspects of a design or process are thoroughly evaluated before the product is produced. This is a proactive approach to anticipate as many potential failures at the earliest possible stage so that changes can be made with little cost involved.

The traditional way of conducting FMEA is through brainstorming. The results of the session are failure analyses and proposed engineering controls, which are recorded manually onto hard copies or into spreadsheets. Hence, FMEA reports contain valuable engineering know-how that could be reused to avoid re-occurrence of similar

*Corresponding author. E-mail: P.C.Teoh/K.Case@lboro.ac.uk

International Journal of Computer Integrated Manufacturing
ISSN 0951-192X print/ISSN 1362-3052 online © 2005 Taylor & Francis Group Ltd
http://www.tandf.co.uk/journals
DOI: &doi_num

failures. However, the traditional approach has a serious setback. The method used to record the FMEA report is not suitable for reuse. When the FMEA grows, the information will be increasingly difficult to find. Eventually, users will prefer to recreate their own FMEA rather than reuse existing knowledge with a risk of repeated failures.

In addition to this, the traditional approach has its limitations in the early stages of design (such as conceptual design). First, it is difficult for traditional FMEA to cope with frequent design changes. Second, at the early design stage, the information may be too imprecise and abstract to be reported in the FMEA. Usually, a FMEA is created after detail design has been firmed up, and high costs are incurred for any subsequent design changes. Hence, FMEA has often become a non value-added activity to a company and may be used merely to satisfy the contractual requirements of customers.

One potential method of alleviating the above problems is to enable automatic knowledge retrieval and report generation in the FMEA. The next section explores at some FMEA and modelling related research to address the FMEA generation issue.

2. Literature review

The representations used in FMEA research are either functional or structural models and both are needed to automate the FMEA process (Hunt *et al*. 1995). A functional model describes the intended function or purpose of a system and consists of two main components: function and behaviour. The function of a system provides the design intent, whereas the behaviour describes how the structure of an artefact achieves its function (Gero *et al*. 1991, Russomanno *et al*. 1993). A function can be decomposed into sub-functions to better understand the design through functional analysis. A structural model is defined as 'the components that make up an artefact and their relationships' (Gero *et al*. 1991, p.193), and refers to the configuration of the product or system. The model contains information on all of the components, entities, sub-processes or sub-systems and the interactions among them. In design, each artefact is created to achieve one or more functions and at the same time one or more artefacts can achieve a function. A mapping between functional and structural models represents these relationships.

Ontologies and taxonomies play important roles. According to Benjamin *et al*. (1994, p.2), an ontology is 'a catalogue of terms used in a domain, the rules governing how those terms are combined to make valid statements about situations in that domain, and the sanctioned inferences that can be made when such statements are used in that domain.' Pragmatically, an ontology can be treated as a domain vocabulary with a very specific grammar, such that every time the vocabulary forms a statement using the grammar, the meaning of the statement must be consistent to all its agents or users. Ontologies are sometimes defined informally as taxonomies of classes and their properties in a knowledge domain (Noy and McGuiness 2001). Ontology is often used as a synonym for taxonomy (Sowa 1991, Benjamin *et al*. 1994). A taxonomy is commonly used as a hierarchical structure defined by 'type' or 'is a' relationships. For example, a car is a type of transportation. Part–whole relationships can also be represented, as in a wheel is a part of a car. In this research, an informal definition is used for the ontology and the taxonomy is treated as a subset that constitutes the ontology.

In the behaviour modelling approach suggested by Eubanks *et al*. (1996, 1997), functions are broken down into smaller sub-functions until a level is reached where they can be directly mapped to a structure model. Each basic unit/part of the structure model is responsible for at least one of the functions. Kmenta *et al*. (1999) proposed the use of this method for process FMEA. The process steps can replace the functions in the functional model, where each basic step is responsible for at least one function. The mapping between functional and structural models can represent the behaviour model. Kmenta and Ishii (1998) suggested a method that could be used to model dynamic behaviour. The dynamic behaviour of a system is represented by changes in function – structure mapping at different points in time known as the 'meta-behaviour'. Wirth *et al*. (1996) suggested a knowledge-based system known as WIFA, to support the FMEA process. WIFA is the German acronym for 'knowledge-based FMEA'. The system provides an information model to build function and structure taxonomies as a library for FMEA knowledge. Each component in the structure taxonomy is linked to at least one function and has an assigned list of failure modes. The components can inherit information from the parent in the taxonomy. A function is defined in terms of a list of verbs and contains information about the function carriers, inputs and outputs of the function.

A variety of approaches have been adopted in using these methods in design. Russomanno *et al*. (1993) and Russomanno (1999) organized functions in terms of a set of functional primitives that describe functionality at an abstract level based on the work of Keuneke (1991) and Sembungamoorthy and Chandrasekaran (1986). Functional primitives were treated as the generic categories into which other specific functions can be grouped.

The application of functional primitives is an essential part of research involving functional reasoning. Representation by functional primitives allows the system to be simulated before detailed designs are considered. Functional primitives represent the highest level of abstraction of the functions. A hierarchy of behaviour segments consisting of state transitions and sub-functions represents how a function is accomplished. A behaviour segment can

be further broken down into more detailed segments forming a hierarchy in the functional model. The decomposed behaviour is then linked to the relevant component in the structural model. Hawkins and Woollons (1998) proposed a graphical modelling method for qualitative reasoning known as the 'role model'. A role model can be used to represent the structural information as well as the conceptual representation of energy domains. The modelling task is carried out based on a deep knowledge approach in which both normal working behaviours and failure causes are considered. Goals are assigned to each component represented by the role model. Hence, a failure mode is defined by the failure to reach the intended goal (or function). By modelling the failure modes, the effects of failures can be propagated to the rest of the design. In state analysis (Ruiz *et al*. 2000), physical components/system/parts are represented by objects. The relationships between the objects are represented by the links between parts in a system block diagram. Each link is assigned a number, a link type, a description and an attribute. The number represents the sequence in which the functions are carried out.

In modelling manufacturing processes Bouti *et al*. (1994) demonstrated the use of integrated definition diagrams (IDEF0) and functional reasoning to automate FMEA for an automated manufacturing system. Generic functions (GFs) were used to describe groups of functions with uniform sets of parameters and common characteristics. A GF is identical to the functional primitive from Russomanno *et al*. (1993) and an analogy to a class in the object-oriented method.

Presenting the FMEA causal relationships in graphical format has an advantage in comparison with the traditional tabular format, as the user can easily comprehend the relationships. The approach was adapted by Lee (1999), in a BN (Bayesian networks or belief nets) model. BN were used to provide probabilistic reasoning for the FMEA model. Each node of the networks represented a variable and each variable was characterised by qualitative values. The fuzzy cognitive map (FCM) approach developed by Pelaez *et al*. (1996) uses a similar approach for the FMEA causal chain, but differs in the model representation formalism. The nodes of an FCM model simply represent the failure states rather than variables. The failure state can be a component state, such as 'valve fails open'. The causal links are represented by fuzzy sets expressed in the form of linguistic strengths.

It is clear then that knowledge in FMEA has been modelled either to enable the automation process, or to support an improved FMEA methodology. Price (1997) and Hughes *et al*. (1999) have used CAD data to represent their structural models. In these approaches, functions are normally defined from first principles. For example, the function 'to generate torque' is used to describe a shaft with a rotational degree of freedom about its axis, instead of a more abstract definition such as 'to turn a gear train'. As the structural model is created using CAD data part details are needed in order to have direct mapping to the defined functions. Very detailed results can be produced if the design contains sufficient information for a detailed simulation. However, unless all the required part models are created, the application of FMEA automation using CAD simulation will not be effective. A part model not only refers to the structural model but also includes the functional model and the failure behaviours that are mapped to the part. Creating a part model may not cause too many problems for electrical systems, as this involves investigating what happens when each wire in the circuit goes open circuit, shorts to ground, or shorts to positive and typical failure modes for the components (Hunt *et al*. 1995). However, for mechanical parts the model can be difficult to define because mechanical systems involve a far wider range of domains, including kinematics, fluid dynamics, statics and dynamics (Hughes *et al*. 1999). The mechanical model created by Hughes *et al*. (1999) only managed to cover the area of kinematics in the mechanical domain. The involvement of CAD data in FMEA generation also prevents involvement in the conceptual design stage, as detailed design is needed to create CAD data for such applications. Instead of a CAD model, most research has used some form of graphical diagrams or object representations for both functional and structural models. The advantage of using the graphical or object-based approach is that less detail is required to create the structural model. This approach is more suitable for an abstract model in conceptual design.

A new approach known as the FMEA generation method (FMAG) is proposed, and prototype software has been created and case studies have been conducted to evaluate the method. This paper elaborates the FMAG method and describes the case studies that have been carried out for the evaluation.

3. Proposed method

FMAG is based on the 'knowledge fragment' approach proposed by Kato *et al*. (2002). Previous failure reports are knowledge fragments that reflect the deliberation, reasoning and experience of experts. Each knowledge fragment by itself does not contribute much to the reasoning process. However, they can be organized to provide meaningful knowledge of the process, and they are highly reusable.

The advantage of this approach is that reasoning can be carried out based on a relatively small amount of information. The models needed for the reasoning are less complex compared to the model-based approach. Models are driven by information assigned to the ontologies rather than basic principles, and can be easily composed based on

simple heuristic rules using shallow knowledge reasoning. Hence, it is a suitable method for reasoning in conceptual design.

The following is a list of terms specifically used to explain the proposed FMAG method:

- Entity An object that forms the primary element of a design artefact or process.
- Operator An object that initiates an interaction with entities in a design artefact or process.
- Operand An object that is at the receiving end of an interaction between entities.
- Property An attribute that represents a characteristic of an entity.
- State The condition/value of a property.
- Function A purpose/intent of a design.
- Generic function A purpose/intent of a design that has been categorised into a generic grouping.
- Behaviour The characteristic or state of a generic function.
- Function unit The smallest unit that represents an interaction between an operator and an operand with a function.
- Model An assembly of operators that serves a design purpose/function.
- Precondition A relationship between the state of an operator and the behaviour that it has generated.
- Postcondition A relationship between a behaviour and the state of an operand, that is the result of the behaviour.
- Failure mode An undesired behaviour of a GF.
- Cause The state of an entity that causes a state change on other entities.
- Effect The state change that results from a cause from another entity.
- Current Control The solutions taken to eliminate or mitigate the effect of a cause or failure mode.
- RPN The 'risks priority number' used as an indication of the risk of a particular failure item.

3.1. *Conceptual model*

In FMAG, a functional diagram (Invention Machine Corporation 2000) is used to represent the conceptual design. The diagram consists of one or more function units. A function unit represents an interaction between an operator, an operand and a function. The operator is an object that initiates an interaction with other entities in a design artefact or process. The operand is an object that is at the receiving end of an interaction between entities. Figure 1 shows an example of a typical functional diagram.

Using a printed circuit board (PCB) conveying process as an example, the PCB is moved from the inlet to the outlet of the conveyor as shown in figure 2. When the inlet sensor senses the PCB, the conveyor belt will be activated. When the PCB reaches the outlet, the outlet sensor will sense the PCB and stop the conveyor. The functional diagram for the process is as shown in figure 3.

3.2. *Object, model and function library*

The entities in FMAG are organized into specific libraries to facilitate reuse. For example, a component library can be created using a class hierarchy. The class hierarchy is used to index the components according to their inherent characteristics. The link between a parent and a child is

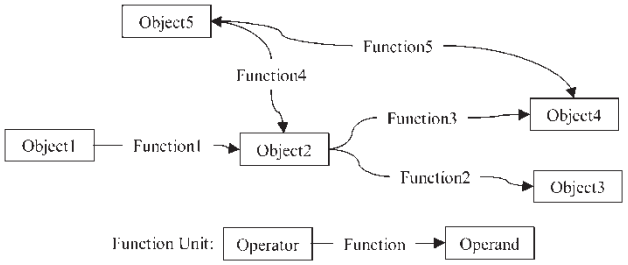

Figure 1. Functional diagram.

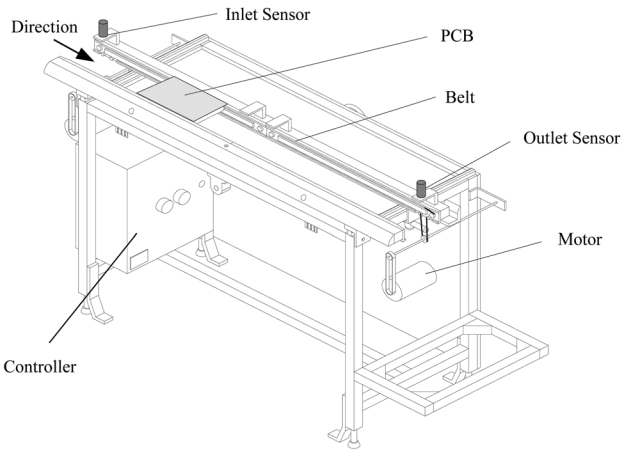

Figure 2. Conveyor system with PCB.

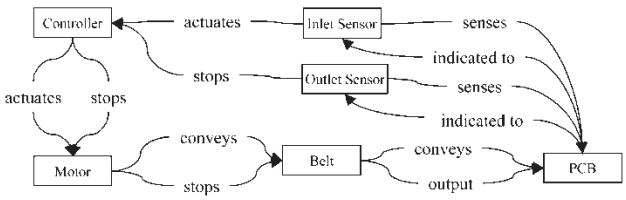

Figure 3. Functional diagram for conveying process.

characterised by the inheritance 'is a' relationship. For example, 'a sensor is a component' as shown in figure 4.

A model is used to organize the components using the assembly tree (figure 5). The assembly tree is used to index the components in terms of their functions. The link between a parent and a child is characterized by the aggregation 'has' relationship. For example, 'a conveyor has sensors'. The conveyor is the model whereas the sensor becomes a component of the model.

Design and process functions are grouped in terms of a generic grouping known as the 'generic functions'. In the FMAG application, the generic function is developed based on the functional basis developed by Hirtz *et al.* (2001). The main groups can then be further divided into lower subgroup functions. Although these groupings are enough to include most of the required functions for the case studies, they are by no means exhaustive. Figure 6 shows an example for a generic function tree.

3.3. Cause and effect propagation

A functional diagram responds to stimulation or changes of state in its components. Causal reasoning drives this response. FMAG divides the knowledge fragment into two parts. They are stored in two separate classes, known as the 'precondition' and the 'postcondition' in the forms of 'operator failure state – failure behaviour' and 'failure behaviour – operand failure state'.

The causal reasoning in FMAG is based on two basic assumptions.

(1) 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.
(2) 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 semantic of the knowledge fragments for the precondition is based on the first assumption, whereas that for the postcondition is based on the second assumption.

The precondition and postcondition gain knowledge through historical data extracted from failure reports and the FMEA. 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

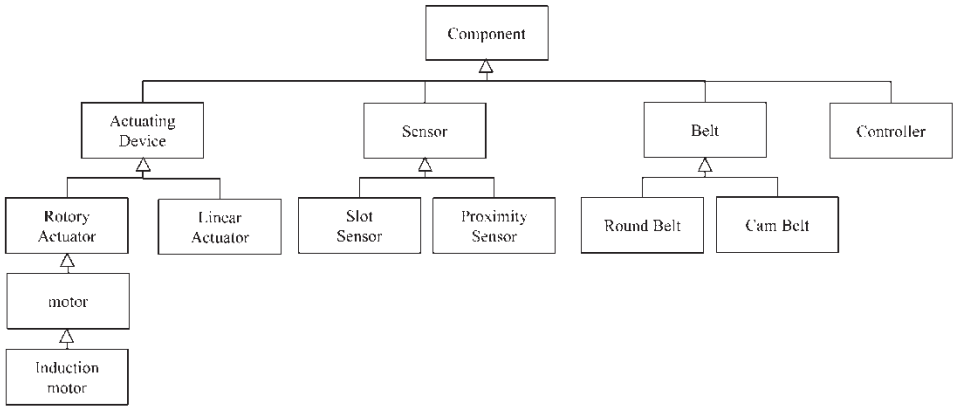

Figure 4. Example of class hierarchy for component library.

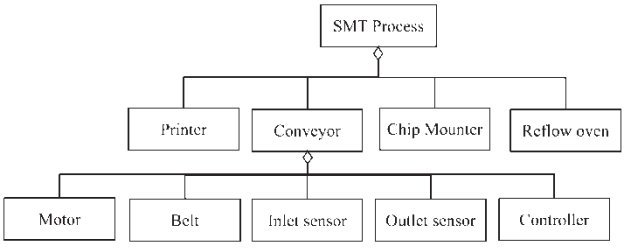

Figure 5. Example of assembly tree for model library.

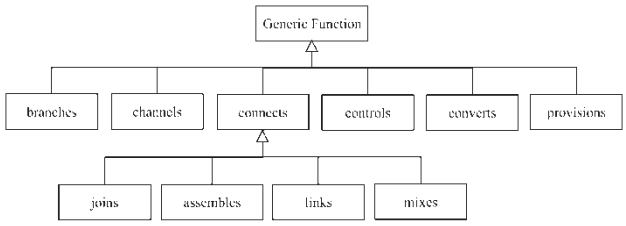

Figure 6. Example of a generic function tree (based on Hirtz *et al.* 2001).

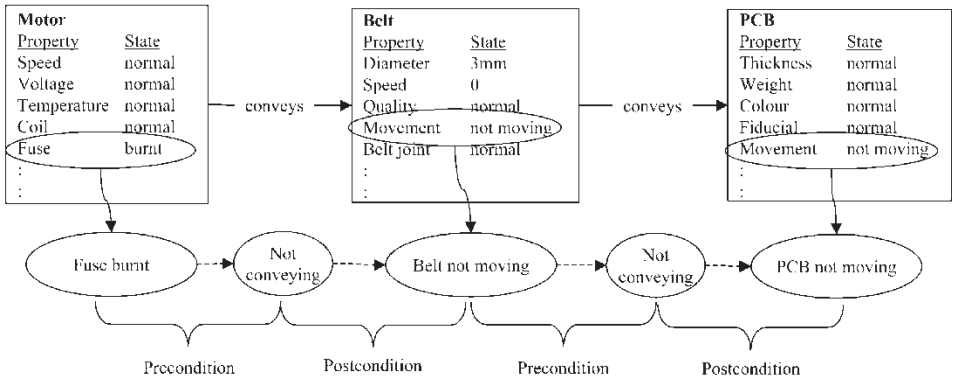

Figure 7. Knowledge fragment capturing.

accumulated events being recorded, precondition and postcondition tables will be formed.

The failure cause and effect is defined by the operator and operand states of a function unit respectively. The failure mode is defined by the failure behaviour of the generic function. In this way, only minimal information is required for the functional and structural models to be used in casual reasoning.

Using this approach, the static knowledge is confined to the entities and their functions, and is excluded from 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 a conveying process (figure 2) as an example, the function of the motor is to move the conveyor belt. The belt in turn is intended to move the PCB that is placed 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. The key information captured from the failure report should be as shown in figure 7.

Hence the knowledge fragment captured in precondition and postcondition tables can be arranged as shown in tables 1 and 2.

The precondition table defines the behaviour of the motor when its fuse is blown and the behaviour of the belt when it is not moving. The postcondition table provides knowledge about the response of the belt when it receives the behaviour 'not conveying' from an operator that is supposed to make the belt move. The postcondition table also provides knowledge about the response of the PCB when it receives the behaviour 'not conveying'. The knowledge is resident in the entities motor, belt and PCB, and not in the function units (as represented by figure 8). This approach provides modularity for the creation of new knowledge.

If a similar or new function unit is created, the operator, operand and the generic function involved can be used as keys to search for the matching states and behaviours in the precondition and postcondition tables. Hence, an entity is able to act or respond to the system through its distinctive 'memory'. Generating the same result with a similar function unit is straightforward. However, there is a possibility that new knowledge can be generated using a new function unit.

Table 1. Precondition table.

Operator	Generic function	Precondition
Motor	Conveys	Fuse burnt – not conveying
Belt	Conveys	Belt not moving – not conveying

Table 2. Postcondition table.

Generic Function	Operand	Postcondition
Conveys	Belt	Not conveying – belt not moving
Conveys	PCB	Not conveying – PCB not moving

Using the same precondition and postcondition tables as above, consider the situation where another designer is creating a design with the new function unit: 'motor conveys PCB'. Assuming that the function unit has never been captured from failure reports, the knowledge will not be available for reasoning under normal circumstances. However, FMAG provides a means to create new knowledge based on possible matching between information in the precondition and postcondition tables.

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 with the name 'PCB' and function 'conveys'. In this case, the likely postcondition (not conveying – PCB not moving) is retrieved. The combination of this information will result in a new case: 'fuse burnt – PCB not moving'. Hence, the PCB has the knowledge to respond to the motor failure even though the case has not previously existed.

3.4. FMEA generation

The causal reasoning technique described in the previous section can be applied throughout the functional diagram. Hence, when a new functional diagram is created for a particular design, the possible failure conditions can be generated based on the historical data saved in the database.

Using a functional diagram, much of the data can be extracted from the causal relationships to form an FMEA item. The user can provide the rest of the information such as the RPN numbers, current control and recommended action at appropriate stages of the FMEA generation process. Figure 9 shows the connection between the functional diagram in figure 3 and the FMEA item for the inlet sensor in the conveyor design.

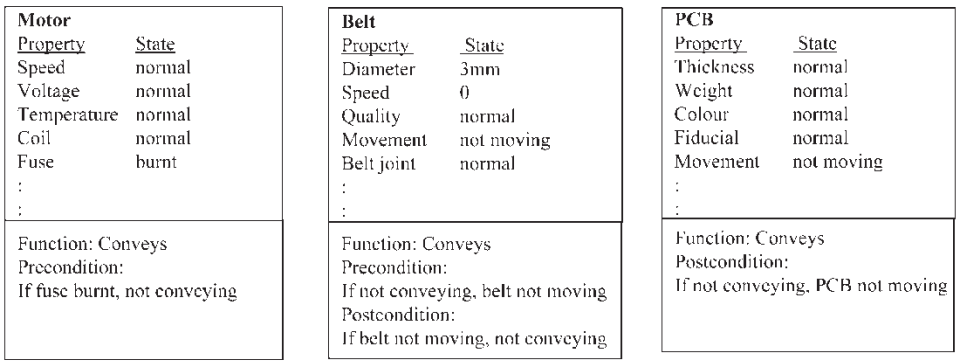

Figure 8. Schematic representation of knowledge fragments in entities.

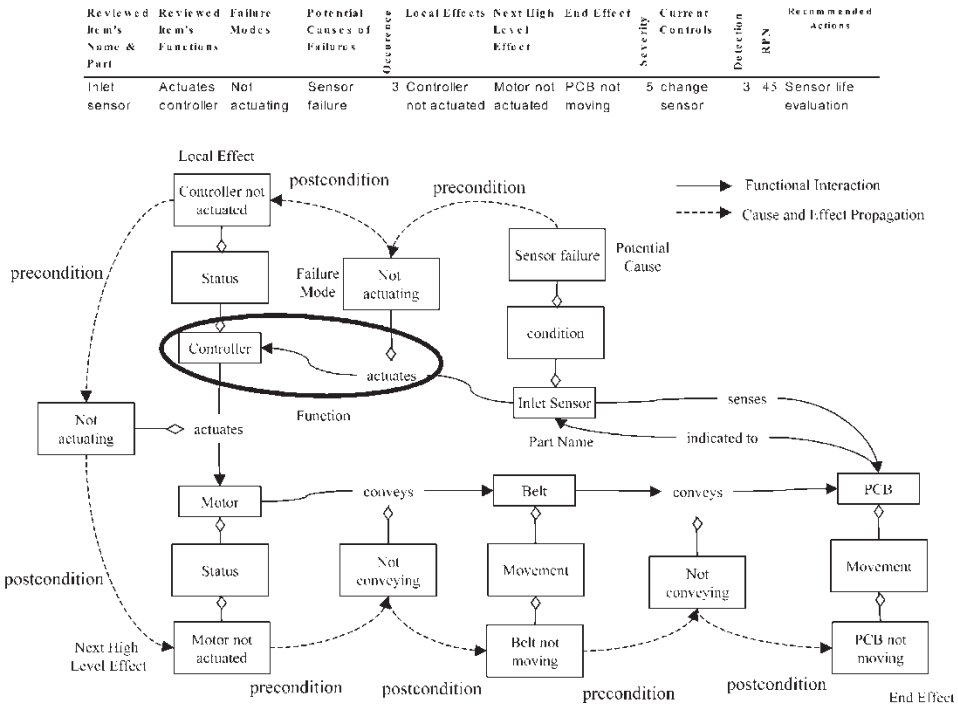

Figure 9. Connection between functional diagram and FMEA.

3.5. *Generated results*

A generated result, as compared to a traditional FMEA created using natural language is as shown in table 3. Table 3 is also an example of a generated FMEA with a complete causal chain. Cause and effect generation can only be possible when historical data exists. If not, additional cause and effects entries may be needed for the rest of the function units until the end of the chain. An entry made just to complete the chain is termed as a 'supportive entry'.

For example in the case of a function unit, 'motor moves belt', the local effect is 'belt not moving'. It is not the end effect as 'belt not moving' can be a cause for 'PCB not moving' in another function unit, 'belt moves PCB'. If the generated FMEA shows the next high-level effect and end effect as 'belt not moving', the chain in said to be incomplete. Hence, the user can create another cause and effect entry for belt moves PCB. This will result in a next high level effect and an end effect of 'PCB not moving'. Since 'PCB not moving' is the end effect for the conveyor system (even though it can be a cause for another machine outside the conveyor system), no further entry is needed.

One way to decide the end effect is to determine the operand of the model. For example, the operand for the model conveyor system is the PCB in a function 'conveyor conveys PCB'. Hence, the PCB can be used as the stopping point of the cause and effect propagation.

Users do not need to complete the solution field for the supportive entry. These items normally will not be used in the FMEA report since they do not provide root causes to the problems. The current control fields in these items are given the values 'NA' (not applicable) during cause and effect entries. These items are termed as 'supportive FMEA items' (table 4).

FMAG allows the components under the same parent in the component library to inherit the generated FMEA from each other. This enhances reuse without the need to recreate similar knowledge. The user can select the generated item that is relevant. Once an inherited FMEA item is selected, the item will appear in the FMEA report in a way that is similar to an item generated in the normal way. For example, the FMEA generated for a slot sensor can be reused in another design that uses another type of sensor through inheritance. The end of the values in the potential causes field for the inherited items is appended with the word 'inherit' to differentiate them from the normal items (table 5).

After FMEA generation, the user can select suitable generated FMEA items to be saved into the FMEA reports. The FMEA reports are kept as separate files so that the user can make any changes to the reports without affecting the contents used for FMEA generation. This is similar to

Table 3. FMAG generated FMEA (top) and traditional FMEA (bottom)

Part/ process step	Part/ process step functions	Potential failure modes	Potential causes	Occurrence	Local effect	Next high level effect	End effect	Severity	Current controls	Detection	RPN
Inlet sensor	Actuate controller	Not actuating	Sensor failure	3	Controller not actuated	Motor not actuated	PCB not moving	5	Change sensor	3	45
Inlet sensor	To activate controller	Fail to activate the controller	Inlet sensor failed to sense the PCB	3	Controller not responding	Motor not running	PCB stuck	5	Use new sensor	3	45

Table 4. Example of a supportive FMEA item.

Part/ process step	Part/ process step functions	Potential failure modes	Potential causes	Occurrence	Local effect	Next high level effect	End effect	Severity	Current controls	Detection	RPN
Inductive motor	Conveys round belt	Sometime not conveying	Motor sometime not running	0	Belt sometime not moving	PCB sometime not moving	PCB sometime not moving	4	NA	0	0

Table 5. Example of an inherited FMEA item.

Part/ process step	Part/ process step functions	Potential failure modes	Potential causes	Occurrence	Local effect	Next high level effect	End effect	Severity	Current controls	Detection	RPN
Reflective sensor	Senses PCB	Sometime not sensing	Sensor dirty (inherit)	3	PCB sometime not sensed	PCB sometime not moving	PCB sometime not moving	4	Clean sensor	4	48

AutoSteve™ (Price 1997) that allows users to customise the generated report.

4. Case studies

Prototype software has been created to evaluate the proposed method, and has been validated using case studies. Two design cases for two-way radio design and three surface mount technology (SMT) process cases have been used in the case studies. The data were based on the failure reports from a factory of Motorola Technology Malaysia.

There is a six-step process for generating a FMEA from a conceptual design:

(1) Define the design or product in terms of a conceptual model. A functional diagram in FMAG will represent the conceptual model. The functional diagram relates the information about the components and functions to the data model before the actual data entry is carried out.
(2) Form objects in the FMAG prototype based on the conceptual model.
(3) Function selection. Under normal circumstances, users do not need to create a new function, as it will be selected from the function library.
(4) Transfer the information in the functional diagram into the software.
(5) Form causes and effects based on previous FMEA or failure reports.
(6) Generate the FMEA items and capture them in the FMEA report.

These steps only serve as a systematic guideline, and need not be followed strictly in sequence to obtain the generated FMEA.

The case studies carried out include:

(1) Case study 1: design FMEA for a two-way radio (model A).
(2) Case study 2: design FMEA for a two-way radio (model B).
(3) Case study 3: process FMEA for a chip placement process.
(4) Case study 4: process FMEA for a solder printing process.
(5) Case study 5: process FMEA for a bare board loading process.

Case study 1 provided accumulated design FMEA knowledge for case study 2, whereas case study 3 to case study 5 demonstrated gradual knowledge accumulation and reuse for process FMEA.

The next section shows the conceptual model for a chip placement process and is typical of the rest of the cases.

4.1. Conceptual model

The purpose of the case studies is to demonstrate the ability of the software to generate a FMEA, and a chip placement process provides a typical example.

Chip mounting is a sub-process of the surface mount technology (SMT) process. One or more chip placement machines carry out the process, and for this case study a simple gantry type chip placement machine is used. Generally, the chip placement machine consists of a gantry system, with a set of nozzles on the gantry head. A conveyor and component feeders are laid on top of the machine tables within the reach envelope of the gantry. A component feeder is used to feed the components to a designated pick-up point in the machine for the pick-and-place process. Cameras are used to locate the PCB and chip components for accurate placement. Each PCB has a set of fiducial marks that enable the machine to calculate the position of the PCB and its solder pads.

During a placement process, the conveyor moves the PCB into a location within the machine envelope. The PCB will be sensed and stopped at the fixed location. The support table underneath the PCB will move up to hold the PCB in place. A PCB camera will be used to locate the fiducial marks of the PCB. The gantry will then move the gantry head to a feeder pick-up point. The nozzles on the gantry head pick the components from the feeder with vacuum force. The component will be brought to a

component camera where positions of the component terminals will be identified. The gantry will then move the head and the nozzle to pre-programmed solder pad locations on the PCB. The nozzle will then place the component by matching the terminals of the component on top of the solder pads. A schematic of the chip placement machine is shown in figure 10. The interactions between the machine components can be extended to a functional diagram as shown in figure 11.

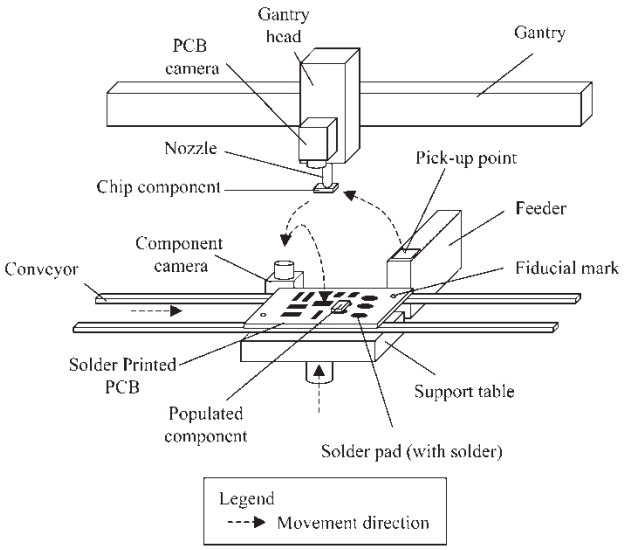

Figure 10. Chip placement machine.

4.2. *Data entry and FMEA generation*

Table 6 provides a summary of the data entries carried out for all five case studies. A total of 224 entries were made based on 163 items from previous FMEA and failure reports.

In FMAG, the generated FMEA items rely on data entry during the cause and effect inputs. However, the number of generated items may not change according to the number of cause and effect inputs. For example, in case study 2, the number of cause and effect inputs was less than case study 1, but the number of generated items was greater. This is due to a high degree of knowledge reuse by case study 2. If the degree of reuse is high, the difference between the generated items and the cause and effect inputs will increase and vice-versa. Hence, the comparison between the generated FMEA items and cause and effect inputs provides an indication of knowledge reuse (figure 12). In figure 12, all the cases have provided a high degree of reuse. Perhaps the most significant reuse was in case study 5 which generated a FMEA without any cause and effect inputs.

The number of FMEA report items created from the FMAG software are usually more than the original reported items. Since the original reported items are the source for the FMEA report in FMAG, the difference between the two

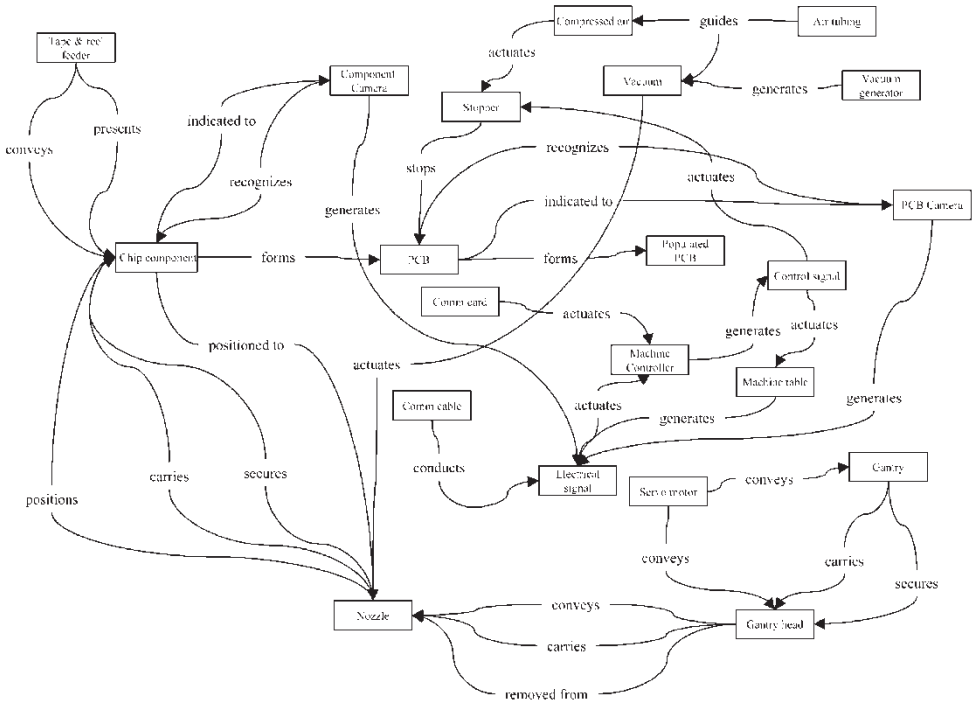

Figure 11. Functional diagram for chip placement process.

Table 6. Data entry summary.

Case Study	1	2	3	4	5	Total
Number of objects	33	14	32	19	15	113
Number of generic functions	14	8	20	14	11	67
Number of function units	48	24	51	30	22	175
Cause and effect inputs	62	54	76	32	0	224
Original reported items	51	46	44	22	0	163

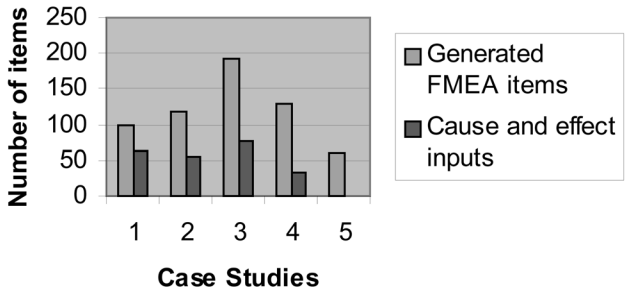

Figure 12. Generated FMEA items and cause and effect inputs comparison.

implies the existence of new knowledge. This is an indication of the creation of new knowledge. Figure 13 provides a comparison between the FMEA report items and original reported items.

Case study 1 did not provide any new knowledge since it is the first model to be created. The rest of the case studies have generated significant amounts of new knowledge.

4.3. Result evaluation

The author and the domain experts from Motorola Technology Malaysia first evaluated the generated results. Two evaluations have been carried out. The first evaluation was to assess the validity of the two basic assumptions that FMAG used to define the causal relationships:

(1) 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.
(2) 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 second evaluation was conducted to validate the generated FMEA based on inherited knowledge from objects in the same families.

Both evaluations also investigated the extent of the generated FMEA used in the FMEA report. This was measured by FMEA utilisation, i.e. the per-centage of items used in FMEA reports against the total generated FMEA items (excluding supportive items).

4.3.1. *Evaluation method.* The author carried out the initial task. The approach involved studying the list of generated results from each case study to obtain the numbers for the following:

(1) Generated FMEA items (non-inherited/inherited) – the total number of rows of the generated FMEA for all the case studies.
(2) Valid FMEA items generated (non-inherited/inherited) – the number of generated FMEA items that the author considered to be valid.
(3) Generated supportive FMEA items that are only used to complete the causal chain. The current control for these items is given the value 'NA'.
(4) Selected FMEA items – the number of generated FMEA items used in all the FMEA reports.

The above information has been used to quantify the evaluated results. The quantified values were normalized in terms of percentages so that the results of the case studies can be compared. The measurements used were:

(1) FMEA validity (non-inherited/inherited) – the percentage of valid FMEA items against the total generated items
(2) FMEA utilization – the percentage of items used in FMEA reports against the total generated FMEA items (excluding the generated supportive items).

The domain experts from Motorola Technology Malaysia later confirmed the FMEA reports obtained from FMAG.

4.3.2. *Results evaluation.* The results for all five case studies are compiled into three separate tables for the total FMEA items generated, the non-inherited items and inherited items. They are as shown in tables 7, 8 and 9.

Figure 14 shows the comparison chart for FMEA validity between non-inherited and inherited items, as well as the comparisons between the case studies.

The valid FMEA items were very high for non-inherited items (97.3% for the overall result). A few case studies

produced 100% validity. This result strongly supported the validity of the two proposed basic assumptions used in FMAG.

The inherited items have a poor score of only 40.9% for the overall result. However, while looking at the trend from case study 1 and 2 (which represents reuse of design FMEA knowledge), or from case study 3 to 5 (which represents reuse of process FMEA knowledge), the upward trends suggested the possibility that the validity of inherited items may increase with an increase in accumulated knowledge.

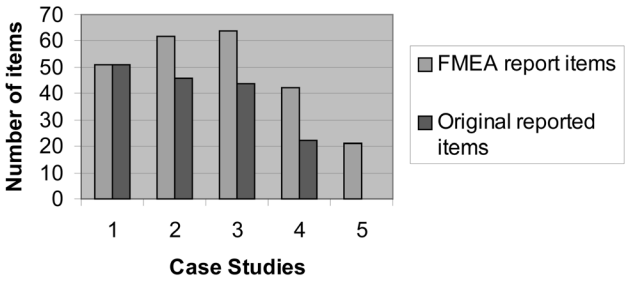

Figure 13. Comparison Between FMEA report items and original reported items.

Figure 15 provides a comparison chart in terms of FMEA utilization between non-inherited and inherited items as well as the comparisons between the case studies.

FMEA utilization was very high for the non-inherited items as compared to the inherited items. This is obvious as there were more valid non-inherited items being generated for the selection. However, looking at the design studies (1 and 2) and the process studies (4, 5 and 6), no trend was found to indicate a relationship between FMEA utilization and the increase in accumulated knowledge. This is because not all valid FMEA items were chosen for inclusion in the FMEA report, as in some cases the selection was subject to the discretion of the user.

The overall result for FMEA validity and utilization were not high due to the poor score for the inherited items. However the contributions of inherited knowledge cannot be disregarded. As shown in case study 5, the number of inherited FMEA items chosen for inclusion in the FMEA

Table 7. Overall evaluation result.

Case Study	1	2	3	4	5	Overall
Generated FMEA items	98	119	191	130	60	598
Valid FMEA items	64	88	135	102	47	436
Supportive FMEA items	20	29	86	58	32	225
FMEA report items	51	62	64	42	21	240
FMEA validity (%)	65.3	73.9	70.7	78.5	78.3	72.9
FMEA utilization (%)	65.4	68.9	61.0	58.3	75.0	64.3

Table 8. Result for non-inherited items.

Case Study	1	2	3	4	5	Overall
Generated FMEA items	60	65	113	76	25	339
Valid FMEA items	60	65	107	73	25	330
Supportive FMEA items	13	11	54	31	13	122
FMEA report items	47	49	54	37	12	199
FMEA validity (%)	100.0	100.0	94.7	96.1	100.0	97.3
FMEA utilization (%)	100.0	90.7	91.5	82.2	100.0	91.7

Table 9. Result for inherited items.

Case Study	1	2	3	4	5	Overall
Generated FMEA items	38	54	78	54	35	259
Valid FMEA items	4	23	28	29	22	106
Supportive FMEA items	7	18	32	27	19	103
FMEA report items	4	13	10	5	9	41
FMEA validity (%)	10.5	42.6	35.9	53.7	62.9	40.9
FMEA utilization (%)	12.9	36.1	21.7	18.5	56.3	26.3

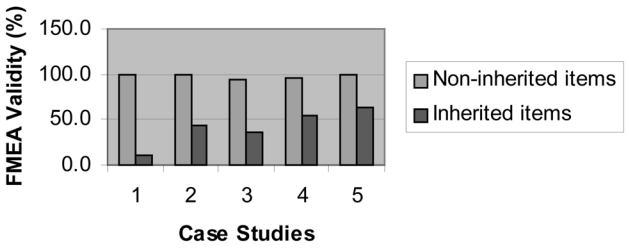

Figure 14. FMEA validity.

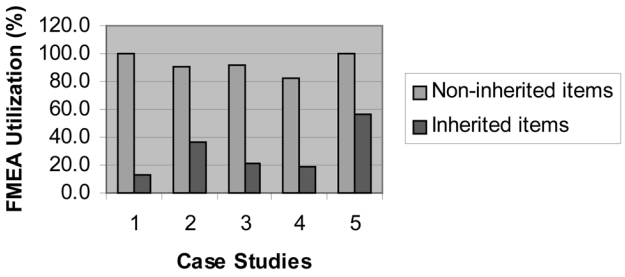

Figure 15. FMEA utilization.

report was 9 out of the total of 21 items (43%). This suggests the possible strong involvement of inherited items in the final report.

The case studies have demonstrated the system's capability of generating a FMEA for a product and process design. They have also shown FMEA generation with incomplete functional information. Both previous FMEA reports and failure reports are equally useful as the sources for FMAG.

The number of function units provides an indication of the complexity of the conceptual model. The difference between the cause and effect inputs and the original reported items implies the amount of additional effort needed in the conceptual modelling process. The comparison between the generated FMEA items and cause and effect inputs provides an indication of knowledge reuse. The comparison between the FMEA report items and original reported items implies the existence of new knowledge.

The number of generated FMEA items and reported items increases as knowledge is reused. The number of supportive items increases with the complexity of a model, and modelling effort increases accordingly. With sufficient historic knowledge, the FMAG software is able to generate an FMEA report without data entry. The high FMEA validity and utilization for the non-inherited items shows that the basic assumptions for FMAG can be used for cause and effect reasoning.

Inherited items were not very helpful in suggesting accurate results for FMEA generation. However, they provide a means for the objects in similar groupings to be reused, and can be used to discover common characteristics among the objects in the group. Hence, this feature is still helpful in supporting knowledge reuse.

5. Discussion

The work to date has demonstrated the usefulness of the general approach, but there are still several outstanding issues arising from limitations deriving from the prototype method. There is clearly a considerable time overhead in populating FMAG with knowledge, which is not experienced with a traditional FMEA approach. The benefit of the automated process can only be realized when a large amount of knowledge is available.

The difference between design and process FMEA in FMAG is not obvious as both use the same form of operator-function-operand knowledge fragment, distinguished only by the use of 'part names' or 'process steps'.

Data inputs from the failure reports need to be supplemented by brainstorming sessions and knowledge sharing with other applications. The functional basis provides necessary standardization for the functions, but it is not sufficient to cover every aspect of the model. Hence, non-standard functions have to be introduced. The implicit functional model in FMAG may cause important information to be missed during the retrieval process. This can be overcome by providing intelligence to the software to search for relevant function units.

The current FMAG prototype has three main limitations: (1) its inability to represent different instances of the same model; (2) to model logical processes; and (3) to represent dynamic behaviours. The first situation could be handled by software improvements to recognize instance names rather than class names while the second could be accommodated by use of scenarios in a layered functional representation.

The FMAG prototype is not able to provide all possible effects propagation as only a single path for each generated local effect is used for propagation. A multiple path propagation algorithm might be feasible but would undoubtedly be complex and computational intensive. Hence model decomposition is seen as a more likely solution.

FMAG cannot rely on isolated data sources and hence needs to share knowledge with other applications, perhaps using a common ontology for requirement analysis and problem diagnosis. The FMAG method would be used to supplement the FMEA generation in detail design, stored in the FMAG database and made available to other applications.

6. Conclusion

This research was prompted by the lack of generic tools to support FMEA report generation in the conceptual design stage. A 'knowledge fragment' approach known as FMAG has been adapted to provide causal reasoning with minimal information input. A conceptual model is organized in term of objects, functions and models in libraries to facilitate reuse. A cause and effect propagation method has been proposed. The method is based on two basic assumptions:

(1) 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.
(2) There exists a functional behaviour where if there is a change to that behaviour, it will cause the correspond operand to change its state accordingly

Five case studies with actual design and manufacturing data, including two design cases and three process cases, have been carried out to validate the FMAG method. The case studies have demonstrated FMEA generation based on user input and the reuse of existing knowledge. The studies were carried out in sequence to portray the trend of knowledge accumulation for FMEA generation. Verifications were carried out to assess the accuracy of the generated results, and confirmed the validity of the FMAG basic assumptions. However, results from the inheritance features need further improvement so that they can be used as a feature to discover common characteristics among objects in the same family.

Hence, the case studies have proved the feasibility of applying FMAG to fulfil the need for a tool to support FMEA report generation during conceptual design. However, there are still issues derived from the prototype that needs further improvement before FMAG can be used in actual working environment.

Acknowledgements

The authors would like to thank the Wolfson School of Mechanical and Manufacturing Engineering, Loughborough University and Motorola Technology Malaysia PLC for supporting the research.

References

Benjamin, P. C., Menzel, C. C., Mayer, R. J., Fillion, F., Futrell, M. T., deWitte, P. S. and Lingineni, M., 1994, IDEF5 Method Report. Information Integration for Concurrent Engineering (IICE). Knowledge Based Systems, Inc.

Bouti, A., Ait K. D. and Dhouib, K., 1994, Automated manufacturing systems failure analysis based on a functional reasoning. *10th ISPE/IFAC International Conference on CAD/CAM, Robotics and Factories of the Future CARs & FOF '94. Information Technology for Modern Manufacturing*. (Kanata, Ontario: OCRI Publications), pp. 423–429.

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

Chen, H. C., 1996, Failure modes and effects analysis training manual, *Personal Communication*, Hen Technology Inc, United States.

Eubanks, C. F., Kmenta, S. and Ishii, K., 1996, System behavior modelling as a basis for advance failure modes and effects analysis. *Proceeding of the ASME Design Engineering Technical Conference and Computers in Engineering Conference*, Irvine, CA, August, 1996.

Eubanks, C. F., Kmenta, S. and Ishii, K., 1997, Advanced failure modes and effects analysis using behavior modelling. *Proceeding of the ASME Design Engineering Technical Conference and Design Theory and Methodology Conference*, Sacramento, CA, 14–17 Sept. 1997

Gero, J. S., Tham, K. W. and Lee, H. S., 1991, Behaviour – a link between functional and structure in design. *Proceeding of the IFIP WG 5.2 Working Conference on Intelligent Computer Aided Design*, Colombus, OH, 30 Sept – 3 Oct. 1991. (Elsevier).

Hawkins, P. G. and Woollons, D. J., 1998, Failure modes and effects analysis of complex engineering systems using functional models. *Artificial Intelligence in Engineering*, **12**(4), 375–397.

Hirtz, J., Stone, R. B., McAdams, D. A., Szykman, S. and Wood, K. L., 2001, A functional basis for engineering design: reconciling and evolving previous efforts. *Research in Engineering Design*, **13**(2), 65–82.

Hughes, N., Chou, E., Price, C. J. and Lee, M., 1999, Automating mechanical FMEA using functional models. *Proceedings of the Twelfth International Florida AI Research Society Conference*, (AAAI Press, Menlo, CA), pp. 394–398.

Hunt, J. E., Pugh, D. R. and Price, C. J., 1995, Failure mode effects analysis: a practical application of functional modelling. *Applied Artificial Intelligence*, **9**(1), 33–44.

Invention Machine Corporation, 2001, TechOptimizer. http://www.invention-machine.com/products/techoptimizer.cfm

Kato, Y., Shirakawa, T. and Hori, K., 2002, Utilizing fault cases for supporting fault diagnosis tasks. Department of Advanced Interdisciplinary Studies, University of Tokyo. http://www.ai.rcast.u-tokyo.ac.jp/~yoshi/papers/kes2002.pdf

Keuneke, A., 1991, Device representation, the significant of functional knowledge. *IEEE Expert*, Vol. **6**(2), 22–25.

Kmenta, S. and Ishii, K., 1998, Advanced FMEA using meta behavior modelling for concurrent design of products and controls. *Proceeding of the ASME Design Engineering Technical Conference*, Atlanta, GA.

Kmenta, S., Fitch, P. and Ishii, K., 1999, Advanced failure modes and effects analysis of complex processes. *Proceeding of the ASME Design Engineering Technical Conferences*, Las Vegas.

Lee, B. H., 1999, Design FMEA for mechatronic systems using Bayesian network causal models. *Proceedings of the ASME Design Engineering Technical Conferences*, **1,** 1235–1246.

Noy, N. F. and McGuinness, D. L., 2001, Ontology development 101: a guide to creating your first ontology. Stanford University, Stanford CA. http://protege.stanford.edu/publications/ontology_development/ontology101.pdf

Pelaez, C. E. and Bowles, J. B., 1996, Using fuzzy cognitive maps as a system model for failure modes and effects analysis. *Information Sciences*, Vol. **88**(1–4), p. 177–199.

PRICE, C. J., 1997, AutoSteve: automated electrical design analysis. *IEE Colloquium (Digest)*, **338,** 4/1–4/3. *Proceedings of the 1997 IEE Colloquium on Applications of Model-Based Reasoning*.

PRICE, C. J., PUGH, D. R., WILSON, M. S. and SNOOKE, N., 1995, FLAME system: automating electrical failure mode & effects analysis (FMEA). *Proceedings of the Annual Reliability and Maintainability Symposium*, pp. 90–95.

RUIZ, I., PANIAGUA, E., ALBERTO, J. and SANABRIA, J., 2000, State analysis: an alternative approach to FMEA, FTA and Markov analysis. *Proceedings of the Annual Reliability and Maintainability Symposium*, pp. 370–375.

RUSSOMANNO, D. J., 1999, A function-centered framework for reasoning about system failure at multiple levels of abstraction. *Expert System*, **16**(3), pp. 148–169.

RUSSOMANNO, D. J., BONNELL, R. D. and BOWLES, J. B., 1993, Functional reasoning in a failure modes and effects analysis (FMEA) expert-system. *Proceedings of the Annual Reliability and Maintenability Symposioum*, pp. 339–347.

SEMBUGAMOORTHY, V. and CHANDERASEKARAN, B., 1986, Functional representation of devices and compilation of diagnositic problem solving systems. In: J. Kolodner and C. Reisbeck (Eds), *Experience, Memory, and Reasoning* (Mahwah, NJ: Lawrence Erlbaum), pp. 47–73.

SOWA, J. F., 1991, *Principles of Semantic Networks – Exploration in the Representation of Knowledge*. *Part 1: Issues in Knowledge Representation*. (New York: Morgan Kaufmann).

WIRTH, R., BERTHOLD, B., KRAMER, A. and GERHARD, P., 1996, Knowledge-based support of system analysis for analysis of failure modes and effects. *Engineering Applied Artificial Intelligence*, **9**(9), 219–229.