Performing FMEA Using Ontologies

Lars Dittmann (lars.dittmann@pim.uni-essen.de)
Tim Rademacher (tim.rademacher@uni-essen.de)
Stephan Zelewski (stephan.zelewski@pim.uni-essen.de)
Institute of Production and Industrial Management; University of Duisburg-Essen, Campus Essen,
Universitätsstraße 9, 45141 Essen, Germany

Abstract

The paper aims to introduce an approach that integrates a technique of knowledge engineering (*Ontologies*) and a technique of quality engineering (*Failure Mode and Effects Analysis*). An approach will be set up that shows the potentials of combining IT-based systems of knowledge and quality engineering. Particularly with regard to the quality engineering technique, the paper aims to demonstrate the advantages of this approach.

Introduction

The management of a firm's knowledge is made difficult mainly by two problems. Firstly, relevant knowledge may often not be found in an explicit form like databases, but in documents like project statements and QM handbooks. Additionally, it is often included in documents referring to a certain circumstance. This implicit knowledge is not immediately accessible; in particular it cannot be acquired by way of a conventional database system. Secondly, the access to knowledge is encumbered with the problem that different actors use different terms to talk about the same topic.

In recent years the Failure Mode and Effects Analysis (FMEA) has become highly relevant in modern quality management systems, due to QS 9000. The FMEA aims to prevent failures in early phases of the product life cycle. That underlies the assumption that the costs for wiping out failures increase exponentially with every phase of the product life cycle, so that an early prevention is much cheaper than late repairs. But the effort to develop an FMEA is mainly considered as high or very high due to the number of involved persons (Stock & Stone & Tumer, 2003). In addition, the advantages that result out of failure prevention can not be perceived immediately. To shorten the process of FMEA development and earning results, the knowledge included in already developed FMEA has to be reused.

The FMEA knowledge reuse suffers from a major shortcoming mentioned by Wirth et al., 1996[1)]: the FMEA-related information is acquired in natural language. The natural language is responsible for further problems. The analyses are hardly reusable because the systematized components, functions and failure modes are not made explicit. Their meaning depends on the interpretation of the team who performs the FMEA and can differ when another team reuses this FMEA, or even if the same team tries to reuse it on a later occasion. Already performed FMEA are hardly *comprehensible*. Caused by the lack of reusability the FMEA is often built from scratch without making use of older FMEA. So they are often *incomplete*. In case of large systems examined in an FMEA, it is barely possible to avoid *inconsistencies*. Especially by fulfilling operational tasks based on strong division of labor, like conducting an FMEA, ontologies can help to integrate task relevant knowledge components by structuring the domain knowledge uniform.

In the relevant literature one will find several approaches to facilitate FMEA performing by information systems techniques or to reuse the knowledge of an FMEA: Forschungsgemeinschaft Qualitätssicherung [FQS], 1994; Nedeß & Nickel, 1992; Struss, 2004; Wirth et al., 1996. But to the knowledge of the authors only Lee (2001) presents an approach bringing ontologies and FMEA explicitly together. Lee's approach suffers from a missing explicit definition of *ontologies* and from not offering a representation language to constitute ontologies. From his point of view inferences seem not to be necessary for ontologies. For Lee an ontology is quite the same as a conceptual model without rules. Furthermore he fails to explain his understanding of an FMEA and to examine the elements of an FMEA. These basic works shall be done in our paper.

FMEA and its major components

The FMEA is a method for improving quality. It should be conducted mostly in early phases of product development. It is a formalized, analytical method (Stockinger, 1989) and its origins lie in the Apollo-Project of the NASA in the sixties of the last century (Deutsche Gesellschaft für Qualität [DGQ], 2001).

The basic idea of the method is to elicit failures of a product already while planning it. At the same time the method can be used to find deficits in the production process systematically and to cure these deficits.

The main goals of an FMEA are (Eberhardt, 2003; Theden, 1997):

- critical components shall be identified to guarantee the constructive and production technical quality of a product,

[1] Indeed, the authors mention two fundamental weaknesses of FMEA: In addition to the natural language, they complain about the non-existence of a methodological guideline on how to conduct an FMEA. Since there are several committees that are standardizing the conducting of an FMEA, this point is omitted (e. g. http://www.aiag.org).

- possible failures shall be identified and localized early to guarantee demanded functions,
- risks shall be estimated,
- change expenditures after starting mass production shall be reduced,
- the utilization and dissemination of knowledge shall be facilitated and
- the support of continuous improvement shall be implemented.

The processes to conduct an FMEA have been standardized in several works, like Verband der Automobilindustrie (1996), Rabbitt and Bergh (1998), DGQ (2001), Society of Automotive Engineers [SAE] (2002) and MIL-STD-1629 (1980).[2)]

Basically five phases (*system structure*, *function structure*, *failure analysis*, *risk analysis* and *optimization*) have to be accomplished. Before these five phases the performing of an FMEA has to be prepared. This consists for example of anchoring the FMEA-thinking and its benefits on the management level and setting up an FMEA-team. After organizational preparation the object of investigation has to be determined specifically.[3)]

The object of investigation is structured and described regarding layout and function respectively (FQS, 1994). All relevant components of a system have to be identified. Physical components as well as process steps can be relevant. All functions have to be "exposed" for each component. Afterwards one conducts a search for potential failure modes, effects and causes for each function. All combinations of failure modes, effects and causes have to be determined. The found failure modes have to be systemized and valued. A risk analysis is conducted. Failures that are considered to have serious impact on a product or process are appointed for containment actions. It has to be examined which kind of failure on a higher component-level can be provoked by and what failure modes on a lower component-level are reasonable for a potential failure.

The realization of containment actions is done by explicit assignment of responsibilities as intention to optimization. After this a verification of the impact of containment actions follows by validating the implemented containment actions.

As supplementary instrument for conducting an FMEA a standardized form is used. A more conceptual view is shown in figure 1.

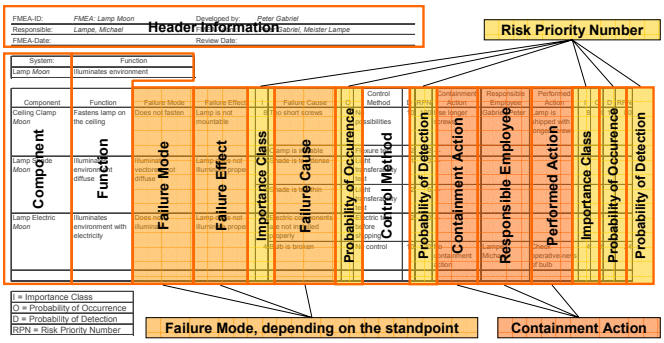

Figure 1: Identification of elements of an FMEA

The following elements can be identified:

- *Header Information*: The FMEA form contains some information about e. g. participating persons, responsible persons and time.
- *Component*: The analyzed system with its components is described.[4)]
- *Function*: Each component has one function minimum. The functions are also described.
- *Failure Mode, Failure Effect and Failure Cause*: Each function has one or multiple failure modes. Each failure mode causes a failure effect and is a result of a failure cause in a more detailed view. Since failure effects and failure causes are failure modes too, depending on the standpoint, they all include the same kind of description.
- *Importance Class, Probability of Occurrence and Probability of Detection*: The importance class represents the importance of a failure in case of occurrence. If a failure is dangerous and can cause harm to the users of a product, the importance class is high. The importance class and the probabilities of occurrence and of detection are integer values ranging from 1 to 10. Their product equals the risk priority number (RPN).
- *Control Method*: A control method is performed to detect a failure mode. The probability of detection strongly depends on the effectiveness of a performed control method.
- *Containment Action, Responsible Employee and Performed Action*: If an RPN's value is too high, a containment action must be defined that can lower the probability of occurrence and/or increase the probability of detection in a concrete case. The name of the responsible employee is given. The performed action describes performed steps to enhance the product in general *in praxi*. As these three elements only occur together, they are assigned to one conceptual element.

Several problem fields need to be taken in account to conduct an FMEA successfully (Nedeß & Nickel, 1992).

The *knowledge supply problem* pertains to the availability of knowledge, the existence of motivated experts or comprehensive information storages. The *knowledge processing*

[2] SAE J1379 is the actual binding standard for utilization of FMEA by the Big Three of the American automotive industry (Daimler-Chrysler, Ford and General Motors). For evolution and contents of SAE J1379 see Carlson & McCullen & Miller (1996).

[3] Many authors stress the importance of these early preparations. Specifically mentioned are for example project organization, setting up a team and training participating employees (cf. e. g. Pfeifer (2002), Schiegg & Viertlböck & Kraus (1999), Witter (1995).

[4] In our approach we do not distinguish between system, subsystem and component, like e. g. SAE (2002). We begin analyzing the deepest level (component) because the other levels can be constituted from this level in the ontology.

problem can be divided in problems to utilize different kinds of knowledge (experience vs. facts), different strategies to solve problems (sequential failure analysis, starting with potential failures vs. one step determination of full failure modes) and multidisciplinarity (enhancing teamwork). The *complexity problem* is connected to reducing the time-consuming effort and the level of difficulty. The *integration problem* tries to connect different fields of an enterprise to get an integrated manufacturing system (to make available the knowledge of quality engineering for the personnel department). To reuse the knowledge of already conducted FMEA it is necessary to give appropriate search criteria (*searching problem*). The *update problem* relates to the necessity of keeping the knowledge in the knowledge base up to date, especially to avoid redundant work.

Ontologies and their major components

The concept *ontologies*, used in our context, stems from the field of artificial intelligence research. The most common definition goes back to Gruber (1993): an ontology is 'an *explicit specification of a conceptualization*'. Based on that, for our context, an ontology is defined as an explicit and formalized specification of the "sensible" linguistic means of expression for a shared conceptualization of real world phenomena, which can be considered as perceivable or imaginable in a subject or purpose dependent section of reality, used or needed for communication between several actors (Zelewski, 2002). An ontology consists of definitions of concepts, relations[5)] and rules and is used in knowledge-based systems with the potential to employ inference. The advantage of an ontology is that it can annul the problems of FMEA reuse.

In this paper, the modeling primitives comprise a tuple O above the universe U :[6)]

$$O = \langle C, R, F \rangle$$

Consisting of:

- $C \subseteq U$ representing the set of concepts.
- $R \subseteq U$ representing the set of relations, with $R = R^H \cup R^R$ and $R^H \cap R^R = \varnothing$.
  R^H comprises the hierarchical terminological relations between concepts and R^R comprises the remaining relations between concepts. It is stated that
  $R^H \subseteq C \times C$ and $R^R \subseteq \bigtimes_{i=1}^{n} C_i$ with $n \geq 2$.
  The hierarchical relations have a special relevance in our context contrary to the remaining relations because the cognitive ability of human abstraction tends to prefer these terminological relations to structure concepts. Especially in our context of taking apart components, the hierarchical relation is used to systematize a product or process.
- The set $F = F_{fix} \cup F_{spec}$ with $F_{fix} \cap F_{spec} = \varnothing$ contains unrestricted predicate calculus formulas consisting of expressions using elements of C and R as unary and binary predicate symbols, respectively, (e.g. $c(x)$ with $c \in C$ and $r(y,z)$ with $r \in R$) and first order logic particles (like negation $\neg$, implication $\rightarrow$ and conjunction $\wedge$) as rules. F_{fix} contains the set of rules that reflect properties (attributes, consistency rules). These rules reflect the semantic of predefined generic predicates as precise as possible. F_{spec} contains rules that are only restricted by the specific domain of the defined ontology. They are needed to explicate implicit knowledge of the knowledge base.

Ontologies are formalized knowledge, represented in a language that supports reasoning. In the case of explicating gaps, the knowledge about failures is only implicitly contained in documents and not explicitly available for information systems, an inference engine allows explicating the implicit knowledge. Using non-deductive inference rules can expand the explicable content of a knowledge base significantly. The user of an ontology-based information system gets a more valuable answer than he would get by just using a common database query. The inference mechanism takes positive effects on quality, actuality, acceptance and trustworthiness of the available knowledge.

Ontologies can support the development and performance of an FMEA in two ways:

- They offer a common understanding of the concepts of a domain (in this case in the domain of FMEA in general and its issued instances). There is no need of interpretation.
- The knowledge held in an ontology is machine-readable and distinct/explicit.

Both advantages support a reuse of FMEA-knowledge: by developing an FMEA the editors have to agree on certain concepts, relations and rules for a domain. Hence, an ontology-based FMEA is *comprehensible*. The common understanding facilitates the reuse so that different FMEA can be used as basis for a new FMEA and to improve *completeness*. Ontology-based FMEA offer rules that can be used to assure *consistency* even in large ontologies.

The FMEA-knowledge implemented in an ontology can facilitate the conduction of an FMEA, as already mentioned. In addition, the ontology can be used for management information systems because it contains knowledge about e. g. products and processes. Use examples of such an ontology-based information system are e. g. fault-tree analyses, diagnostic systems and employees' skills management. In return an integrated ontology-based management information system can facilitate the effort to implement FMEA because the existing knowledge can be reused easily.

[5] Some authors refer to relations as attributes, properties or methods (Erdmann, 2001).

[6] See for different formalized definitions Erdmann (2001) and Maedche & Staab (2001).

Ontology Development

Our intended ontology-based information system can be explained by the modeling framework shown in figure 2.[7] An application area becomes a mental model through the perceptual experience of a developer. The developer formalizes his mental model using different levels of abstraction. The formalized knowledge as implementation is running inside a computer.

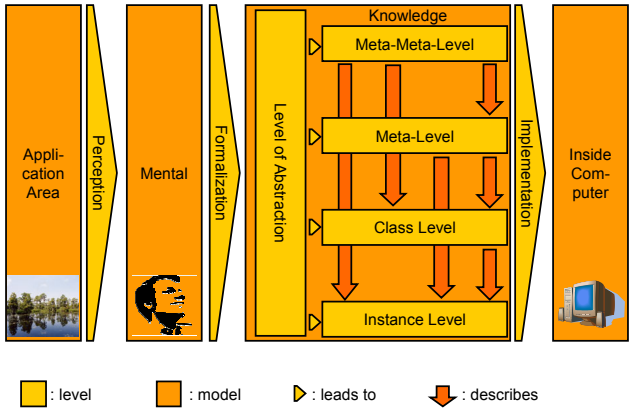

Figure 2: Modeling framework for knowledge based system

One possibility of developing an ontology is the top-down approach.[8] Objective of this approach is the level of abstraction, beginning at the meta-meta-level and ending at the instance level. Every level influences each level of lower abstraction, e. g. the results of the meta-meta-level are used at the following meta-level and also at class and instance level. These levels are taken from the modeling framework (compare figure 2) and used as a simple process model for our ontology development.

Meta-Meta-Level

During the meta-meta-level (the level above the meta-level) phase the foundation of an ontology is defined. In particular, this includes the decision about the used representation language and a definition of its modeling primitives.

Several languages exist for the purpose of ontology representation and for supporting inferencing. F-Logic ("F" stands for "Frames"), which is used in this approach, is the result of combining elements of conceptual modeling with first order logic languages.[9] So called F-atoms that represent entities comparable to classes and relations in the object-oriented (OO) modeling extend the common, only predicate-based approaches. F-atoms can be concepts (like classes in OO-design) and relations (like attributes, relations and methods in OO-design). Additional advantages are that the used code is easy to read and concise as well as that the construction of F-Logic code is easier and more intuitive than other languages (Friedland & Allen, 2003).

F-Logic is able to represent the entities of an ontology as they are:

The concept *c* is linked to the concept *t* by the relation *rel*, in the first statement a single-valued and in the second a multi-valued relation:

```
c[rel=>t]
c[rel=>>t]
```

A special relation is the definition of a subconcept. Concept c_1 is subconcept of c_2 and inherits all of c_2's relations.

```
c1::c2
```

The instance *i* of the concept *c* is defined as follows:

```
i:c
```

The instance *i* has a relation *rel* to a result instance *r* or, in the latter case, a multi-valued relation *rel* to the result instances $r_1, r_2, \ldots, r_n$.

```
i[rel->r]
i[rel->>{r1,r2,…,rn}]
```

The ontology has to represent knowledge about the FMEA itself but also about its included knowledge. The first is represented through the concepts and relations of the ontology while the latter mentioned are represented on the instance level. The aim is to enable or to simplify the reuse of knowledge that is included in FMEA already performed. By choosing a frame-based language like F-Logic, it is possible to assign kinds of knowledge to specific concepts and to represent this assignment explicitly in the used language.

Meta-Level

At the meta-level the key concepts and their relations must be defined. To develop an adequate ontology it is necessary that all included knowledge in an FMEA form is represented in an ontology-based FMEA knowledge base so that transfer back from the ontology into an FMEA form is possible without data loss: The ontology must be data preserving.

Bringing the ontology in line with the standardized components facilitates to achieve a broad acceptance. Thus, the identified components of an FMEA are used in designing the key concepts.

- Following the idea of an ontology, all concepts are derived from the abstract *ROOT_CONCEPT*, the most general entity of the domain. The relations between the concepts are displayed in figure 3, while only relations to other concepts of the ontology are shown in the figure. Relations to concepts like *string* or *integer*, that represent "flat" data types, are omitted due to clarity reasons.

```
/* The concept fmea is a subconcept of
   ROOT_CONCEPT.*/
fmea::ROOT_CONCEPT.
```

- *Fmea*: To store the header information, as mentioned above, of an FMEA form, a concept called *fmea* contains

[7] Many similar depictions in literature exist for a framework to develop information systems (e. g. LITTO, 2002). Note that in our framework the application area is also seen as a model, so there is no reality *a priori*.

[8] Another top-down approach is the Enterprise Model of Uschold & King (1995).

[9] Cf. e. g. Kifer & Lausen & Wu (1995) and Angele & Lausen (2004) for further information on F-Logic.

its information. For instance the responsible person, the FMEA development team and the date of performing the FMEA are related to this concept. An FMEA examines a component or system, thus a relation *examines_component* is part of the concept FMEA as well.

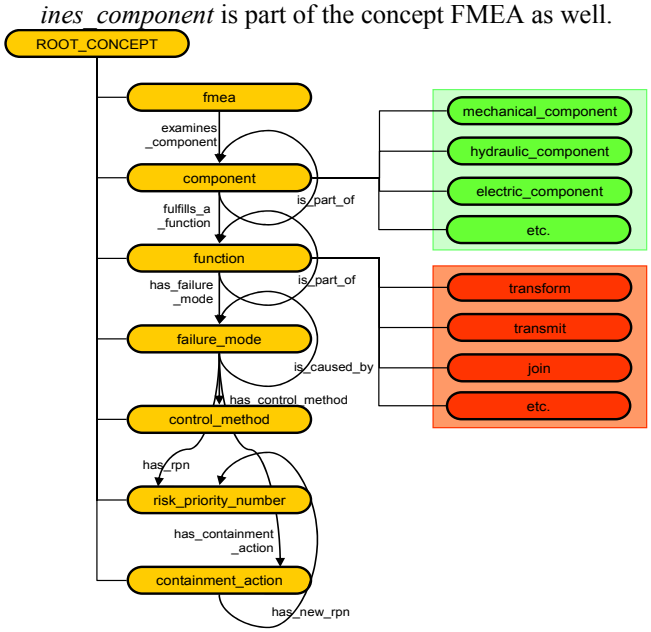

Figure 3: Main Concepts and relations in the FMEA domain

- *Component*: The component describes a complete system, a subsystem or an indivisible element of a system. For *description* purposes a string is related to this concept. Every instance of this concept can include an *is_part_of* relation to another component. This is needed to span a system tree of the product and/or process. Furthermore each component fulfills one or multiple functions (*fulfills_a_function*).

```
/* The concept fmea inherits the following rela-
   tions. The relation development_team relates
   to a set of the co-domain of all instances of
   the concept persons, while the others relate
   to just one instance of their accordant co-
   domains.*/
fmea[ examines_component=>component;
      responsible_person=>person;
      development_team=>>person;
      date_of_performance=>date].
```

- *Function*: The *function* denotes a task that a component has to fulfill. To describe a function, a relation to a string is offered as well. Each function can be part of another function (*is_part_of*). A failure occurs if a function does not fulfill its task. All possible failures of a function are related to this function through *has_failure_mode*.
- *Failure_mode*: According to the FMEA, each failure mode has a cause and an effect. They can be conceived as a triple of failure modes because it depends on the standpoint whether a failure mode is failure mode, effect or cause. The cause is assigned through the *is_caused_by*-relation. The effect is represented by the inverse relation *causes* (that is surveyed later). A failure mode description can be stored in a related string. If, and only if, a failure mode causes another failure mode, which interferes with a function mentioned in an FMEA, the components of a risk priority number can be assigned (*has_rpn*).[10] In this case, a full failure mode triple exists. The same restriction must be applied to the relation *has_control_method*. This relates to a control method that is used to detect failure modes.
- *Risk_priority_number*: This concept exists to store the three factors whose product is called RPN. Four relations to the concept *integer* support access to importance class, to probabilities of occurrence and detection and to their product, the RPN (*importance_class*, *probability_of_occurrence*, *probability_of_detection* and *rpn*).
- *Control_method*: This concept describes the way of controlling a component to find out if a failure has occurred. Thus a relation to a string is offered.
- *Containment_action*: If an RPN is of higher value, the FMEA performer can determine a containment action to prevent the failure. This concept is related to a string-based description of the action that is needed to prevent the failure in a concrete case, a string-relation to the name of the responsible analyst (or better a relation to a concept that represents persons, if the knowledge of e. g. who performed an FMEA is helpful in other ontologies; the concept person of this ontology can be matched to the corresponding concept of the other ontology[11]) and another string-based relation to the performed action that is used to prevent the failure mode in general. If a containment action is determined, a new RPN is calculated that reflects the changes of the containment action (*has_new_rpn*).

Besides these key concepts, rules should be added to explicate implicit knowledge and to assure consistence. Some possible rules cover inverseness, transitiveness and symmetry of relations between concepts.

Inverse relations describe a relation backwards: e. g. the relation *fulfills_function* relates the concept *component* to the concept *function*. In a retrospective view the relation *is_fulfilled_by* that relates the concept *function* with the concept *component* expresses the same. By adding a rule that describes an inverse relation, the knowledge base is enlarged: implicitly included knowledge is made explicit.

```
/* The following example shows the variables Com-
   ponent and Function. The variable Component
   must include an instance of the conceptcompo-
   nent; the variable Function must include an in-
   stance of the concept function. If a component
   fulfills (fulfills_function) a function, one
   can also say that a function is_fulfilled_by a
   component. */
FORALL Component,Function (Component:component)
[#fulfills_function->>Function] <->
       (Function:function)
       [is_fulfilled_by->>Component].
```

[10] The restriction is necessary, because in this case the failure is part of only one row in the FMEA form. As a result a RPN can unequivocally relate to one failure-cause-effect connection.

[11] A survey of methodologies of merging ontologies is given by Gómez-Pérez & Fernández-López & Corcho (2004).

Other examples of inverse relations are:

- A function has a failure mode (*has_failure_mode*), so each failure mode interferes with a function (*interferes_function*).
- A failure mode is caused by (*is_caused_by*) another failure mode. The inverse relation can express that the second failure mode causes (*causes*) the first one.

Transitive relations express the transitiveness of a relation.

```
/* If a first component is part of a second compo-
   nent, and this second component is part of a
   third component, then the first component is
   also part of the third component. */
FORALL X,Y,Z (X:component)[is_part_of->>Z] <-
        (X:component)[is_part_of->>Y] AND
        (Y:component)[is_part_of->>Z].
```

The example above is limited helpful. If a query should determine which components are *direct* parts of another component this transitive rule would make a reality corresponding result impossible.

Symmetric relations are needed if a relation is similar fulfilled in both directions.

```
/* If a component has a sibling, one can certainly
   claim that this component also has a sibling.*/
FORALL X,Y ((X:component)[has_sibling->>Y]) <->
        (Y:component)[has_sibling->>X].
```

In addition to these standard rule types, special rules for the domain of FMEA have to be identified. One of those rules uncovered in the meta-level phase shall be given as an example. The rule is related to the concept *risk_priority_number*. If all single values of the RPN are given, the product can be rule-based calculated.

```
FORALL Importance,Occurrence,Detection,RPN,X
(X:risk_priority_number)[has_rpn->RPN] <-
        (X[importance_class->Importance]) AND
        (X[probability_of_occurrence->Occurrence])
        AND
        (X[probability_of_detection->Detection])
        AND
        (RPN is
                (Importance * Occurrence
                * Detection)
        ).
```

[12]

Class Level

The class level is used to allow a more specific description of the knowledge. As the key concepts are already defined in the meta-level phase, in this phase specific subconcepts of the key concepts are defined.

In order to describe functions and components of an FMEA more precisely and to find similarities between new and already analyzed products, processes and their functions, multiple concepts are derived from the concepts *component* and *function*. As indicated in the boxes in figure 3 on the right-hand, a taxonomy of possible components and another taxonomy of functions are derived from the concept component and from the concept function respectively.

Predefined taxonomies can be used, like the function taxonomy of Birkhofer (1980) or its adaptation by Wirth et al. (1996). Birkhofer developed a taxonomy with 222 verbs that can represent the functions of technical systems. If there is no appropriate taxonomy available, a customized new taxonomy can easily be built application dependent and sufficient for the expected use.

The major advantage of concepts of this level is that they offer the possibility to enhance reuse of old FMEA. In case of developing a new FMEA, the developer has to decide of which kind of component the considered component is. The kind must be available in the component-taxonomy and the considered component is instanced from this concept. By "mounting" a component in the component-tree, similarities can be exposed.

Instance Level

The most specific phase is the instance level. Instances represent the knowledge of real FMEA forms in the devised ontology above.

As already mentioned, the knowledge of an FMEA form is classified in several types. Each type has a matching concept in the ontology so that the knowledge can be transferred directly.

In figure 4, some exemplary instances of concepts of FMEA ontology are shown which represent the knowledge of the FMEA form. Every instance has a unique identifier (e. g. "FMEA Lamp Moon"[13]) which makes it accessible.

The concepts from which the instances are derived are shown in parentheses. Components and functions of an FMEA are not direct instances of the concepts *component* and *function* respectively, but instances of derived subconcepts. Similarities can be revealed through these taxonomies by the derivation.

To clarify this, imagine the following example: A company of the lamp industry is constructing a lamp called "Moon". To achieve high quality, their employees perform an ontology-based FMEA. At first they have to identify all parts of the lamp.

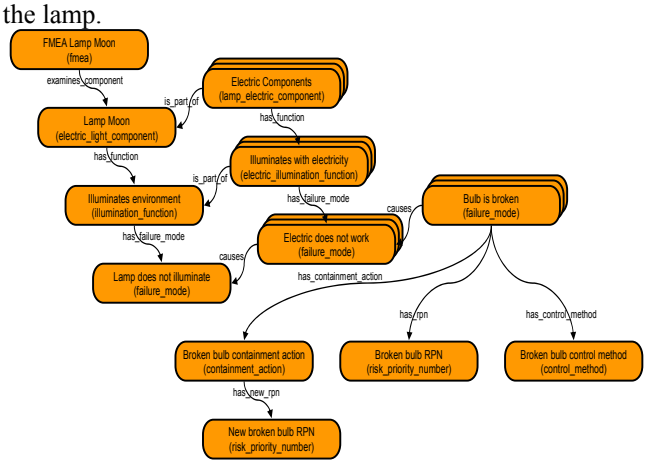

Figure 4: Instances of an ontology-based FMEA

[12] The shown comparison is not part of F-Logic but a built-in extension of the used inference engine Ontobroker by Ontoprise GmbH (http://www.ontoprise.com).

[13] In this example names are given to identify instances. In case of a large ontology it may be helpful to introduce serial numbers for instance identification.

Thus they create the instance Lamp Moon for their lamp. They derive this instance from a subconcept of component: electric_lighting_component. Then the ontology can be queried about other lamps and their components so that the analysts can reuse the construction-knowledge of other lamps. In a similar manner analogies between functions are processed.

Exemplary Queries

The following examples show some possible queries in natural language and in F-Logic that could be helpful for data retrieval during the FMEA process. In this paper they shall make clear the benefit of our approach:

Find all instances of components or a subconcept of component that are part of any instance of the concept *electric_light_component*.

```
FORALL Concept,Subcomponent,Component <-
        Subcomponent [is_part_of->>Component] AND
        Component:electric_light_component AND
        Subcomponent:Concept.
```

Find all instances of the concept component that are part of any instance of the concept *electric_light_component*.

```
FORALL Subcomponent,Component <-
        Subcomponent[is_part_of->>Component] AND
        Component:electric_light_component.
```

Find all instances of the concept function that are functions of any instance of the concept *electric_light_component*.

```
FORALL Function,Component <-
        Function:function AND
        Function[is_fulfilled_by->>Component] AND
        Component:electric_light_component.
```

Find all instances of the concept *failure_mode* that are failure modes of functions of instance *Lamp Moon*.

```
FORALL Mode,Function <-
        Mode:failure_mode AND
        Mode[interferes_function->>Function] AND
        Function:function AND
        Function[is_fulfilled_by->>
        lamp_moon:electric_lighting_component]].
```

Managing the problem fields of an FMEA

Since an ontology and its instances include all knowledge of already performed FMEA and offer the possibility to access this knowledge, the *knowledge supply problem* can be better solved than with ordinary computer-based FMEA conducting. Even if motivated experts are not available, the ontology can explicate implicit knowledge and helpful hints.
The *knowledge processing problem* implicates in general the "soft" facts of conducting an FMEA in collaboration. The ontology-based FMEA development helps to "cure" linguistic divergences. Therefore it enhances the *knowledge processing*.
The *complexity problem* is reduced by using just one knowledge base that is pursuable. The time-consuming effort and the level of difficulty are lowered by the precise systematization of the domain knowledge. E. g. by providing comprehensive concepts for functions the user has the possibility to reuse this knowledge and utilize this systematization.
The ontology-based FMEA offers a possibility to build an integrated manufacturing system by connecting different fields of an enterprise. E. g. knowledge about functions and their failure modes can be used in a diagnostic system. This may solve the mentioned *integration problem*.
The *update problem* is also solved by offering one comprehensive knowledge based system. In particular the possibility to explicate implicit knowledge provides the chance to avoid redundant work in a way that is impossible in conventional knowledge bases.
To solve the *searching problem* an ontology offers a powerful way to gather information. Complex queries are possible.

Conclusio and further research

Even after more than 20 years of application the FMEA is still known as complicated and expensive due to several factors. The paper aims to introduce an approach that shows the possibilities of reducing problem fields which exist by conducting an FMEA.
The technique of *ontologies* is able to facilitate the FMEA proceeding. It can solve the main shortcomings and the resulting problems as mentioned.
But there are still areas that need further research. A commonsense ontology is needed that provides parts of standardized components and a function taxonomy. It would be helpful to base on common consent of industry and technical research. The integration of FMEA into existing knowledge bases has not been examined exhaustively. The authors hope that the methodology OntoClean (Guarino & Welty, 2002) will give good advice for this intend. If an evaluating phase will give a positive result, the methodology shall be integrated in our approach.
In fact, a more finely graded process model is under development at the Institute of Production and Industrial Information Management at the University of Duisburg-Essen. In future it is planned to examine the possibilities of using such ontologies for ontology-based skills management systems. Ontology-based skills management systems are also under investigation at the institute. The naming of responsible employees and organizations offers promising possibilities.

Acknowledgments

This research project is sponsored by the Federal Ministry of Education and Research in the program "Research for the Production of Tomorrow" (Government Aid 02 PD 1060). The project is supervised by Projektträger Produktion und Fertigungstechnologien (PFT), Forschungszentrum Karlsruhe GmbH. The authors would like to thank for the generous support.

References

Angele, J., & Lausen, G. (2004). Ontologies in F-Logic. In Staab, S., & Studer, R. (eds.), *Handbook on Ontologies* (29-50). Berlin: Springer.

Birkhofer, H. (1980). *Analysis and Synthesis of Functions of Technical Products* (in German). Düsseldorf: VDI.

Carlson, W. D., & McCullen, L. R., & Miller, G. H. (1996). Why SAE J1739? *SAE Transactions*, 104, 481-488.

Deutsche Gesellschaft für Qualität (2001). *13-11: FMEA – Failure Mode and Effects Analysis* (in German, 2nd ed.). Berlin: Beuth.

Eberhardt, O. (2003). *Imperilment Analysis with FMEA: The Failure Mode and Effects in Accordance with VDA-Guideline* (in German). Renningen: Expert.

Erdmann, M. (2001). *Ontologies for conceptualistic Modelling of the Semantic of XML* (in German). Doctoral Dissertation, Department of Business Economics, University of Karlsruhe, Karlsruhe: University Press.

Forschungsgemeinschaft Qualitätssicherung (1994). *85-02: Computerized, Knowledge-based Construction of Failure Mode and Effects Analysis* (in German). Berlin: Beuth.

Friedland, N. S., & Allen, P. G. (2003). The Halo Pilot: Towards A digital Aristotle. Retrieved June 9, 2004, from http://www.projecthalo.com/content/docs/halopilot_vulcan_finalreport.pdf.

Gómez-Pérez, A., & Fernández-López, M., & Corcho O. (2004). *Ontological Engineering*. London: Springer Press.

Gruber, T. R. (1993). A Translation Approach to Portable Ontology Specifications. *Knowledge Acquisition*, 5, 199-220.

Guarino, N., & Welty, C. (2002). Evaluating Ontological Decisions with OntoClean. *Communications of the Association for Computing Machinery*, 45, No. 2, 61-65.

Kifer, M., & Lausen, G., & Wu, J. (1995). Logical Foundations of Object-Oriented and Frame-Based Languages. *Journal of the Association for Computing Machinery,* 42, 741-843.

Lee, B. H. (2001): Using FMEA models and ontologies to build diagnostic models. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 15, 281-293.

Litto, M. (2002). *Fault Information System – Information Model and Building Systematics* (in German). Doctoral Dissertation, Institute for Control Engineering of Machine Tools and Manufacturing Units, University of Stuttgart, Heimsheim: Jost-Jetter.

Maedche, A., & Staab, S. (2001). Ontology Learning for the Semantic Web. *IEEE Intelligent Systems*, 16, No. 2, 72-79.

MIL-STD-1629 (1980). *Revision A: Procedures for performing FMECA*. Washington: Author.

Nedeß, C., & Nickel, J. (1992). Knowledge-Based FMEA-Construction (in German). In A.-W. Scheer (Ed.), *Simultaneous Product Development* (in German, 278-333). Munich: gfmt.

Pfeifer, T. (2002). *Quality Management – Strategies, Methods, Techniques*. Munich: Hanser.

Rabbitt, J. T., & Bergh, P. A. (1998). *The QS-9000 book: The fast track to compliance*, New York: Amacom.

Society of Automotive Engineers (2002). SAE J1379: Potential Failure Mode and Effects Analysis in Design (Design FMEA) and Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes (Process FMEA) and Effects Analysis for Machinery (Machinery FMEA). In Society of Automotive Engineers (Eds.), *SAE Handbook* (2003, pp. 1-57). Warrendale: Author.

Schiegg, H., & Viertlböck, M., & Kraus, T. (1999). Process Supporting and Early – System Product FMEA with Objective Codes (in German). *QZ – Qualität und Zuverlässigkeit*, 44, 879-884.

Stock, M. E., & Stone, R. B., & Tumer, I. Y. (2003). *Going back in time to improve design: The elemental function-failure design method*. Proceedings of DETC'2003, Chicago. Retrieved June 29, 2004 from http://function.basiceng.umr.edu/delabsite/publications/conferences/DTM48633.pdf.

Stockinger, K. (1989). FMEA – A Progress Report (in German). *QZ – Qualität und Zuverlässigkeit*, 34, 155-158.

Struss, P. (2004). Model-Based Systems and Qualitative Modelling (in German). In G. Görz & C.-R. Rollinger & Schneeberger, J. (Eds.), *Handbook on Artificial Intelligence* (in German, 4th ed., 431-490). Munich: Oldenbourg.

Theden, P. (1997). *Rentability Analysis of Quality Techniques* (in German), Doctoral Dissertation, Fraunhofer Gesellschaft/IPK, Technical University of Berlin, Berlin: IPK.

Uschold, M., & King, M. (1995). *Towards a Methodology for Building Ontologies* (AIAI Technical Report 183). Edingurgh: University of Edinburgh, AI Applications Institute.

Verband der Automobilindustrie (1996). *Quality Assurance before Mass Production – System-FMEA* (in German, VDA-Band 4, Part 2). Frankfurt: Author.

Wirth, R., & Berthold, B., & Krämer, A., & Peter, G. (1996). Knowledge-Based Support of System Analysis for Failure Mode and Effects Analysis. *Engineering Applications of Artificial Intelligence*, 9, 219-229.

Witter, A. (1995). Development of a Model for Optimized Utilization of the Knowledge Potential of a Process-FMEA (in German). Düsseldorf: VDI.

Zelewski, S. (2002). Organized Experience – Knowledge Management with Ontologies (in German). *Essener Unikate,* No. 18, 63-73.