Expert Systems with Applications 37 (2010) 671–677

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

# An ontology approach to support FMEA studies

V. Ebrahimipour *, K. Rezaie, S. Shokravi

*Department of Industrial Engineering and Center of Excellence for Intelligent Experimental Mechanics, Faculty of Engineering, University of Tehran, P.O. Box 11365-4563, Tehran, Iran*

ARTICLE INFO

*Keywords:*
FMEA
OWL
Protégé
RDF
XML

ABSTRACT

FMEA (Failure Modes and Effects Analysis) is a method to analyze potential reliability problems in the development cycle of the project, making it easier to take actions to overcome such issues, thus enhancing the reliability through design. FMEA is used to identify actions to mitigate the analyzed potential failure modes and their effect on the operations. Anticipating these failure modes, being the central step in the analysis, needs to be carried on extensively, in order to prepare a list of maximum potential failure modes. However, the information stored in risk assessment tools is in the form of textual natural language descriptions that limit computer-based extraction of knowledge for the reuse of the FMEA analyses in other designs or during plant operation. To overcome the limitations of text-based descriptions, FMEA ontology has been proposed that provides a basic set of standard concepts and terms. The development of the ontology uses an upper ontology based on ISO-15926, which defines general-purpose terms and act as a foundation for more specific domains. The ontology is developed so that engineers can build new concepts from the basic set of concepts. This paper evaluates the proposed ontology by means of use cases that measure the performance in finding relevant information used and produced during the safety analyses. In particular, the extraction of knowledge is performed using JTP (an object oriented Modular Reasoning System) that is used for querying the ontology.



## 1. Introduction

Risk management is a central part of any organization's strategic management (Condamin, Louisot, & Naim, 2007). It is the process whereby organizations methodically address the risks attached to their activities with the goal of achieving sustained benefit within each activity and across the portfolio of all activities. The focus of good risk management is the identification and treatment of these risks. Its objective is to add maximum sustainable value to all the activities of the organization. It marshals the understanding of the potential upside and downside of all those factors which can affect the organization. It increases the probability of success, and reduces both the probability of failure and the uncertainty of achieving the organization's overall objectives.

Failure Mode and Effect Analysis (FMEA) is yet another powerful tool used by system safety and reliability engineers/analysts to identify critical parts, functions and components whose failure will lead to undesirable outcomes such as production loss, injury or even an accident. The tool was first proposed by NASA in year 1963 for their obvious reliability requirements (NASA, 1999). Since then, it has been extensively used as a powerful technique for system safety and reliability analysis of products and processes in wide range of industries—particularly aerospace, nuclear, automotive and medical (Rezaie, Amalnik, Gereie, Ostadi, & Shakhseniaee, 2007; Rezaie, Gereie, Ostadi, & Shakhseniaee, 2008). The main objective of FMEA is to discover and prioritize the potential failure modes (by computing respective RPN), which pose a detrimental effect on the system and its performance. The results of the analysis help managers and engineers to identify the failure modes, their causes and correct them during the stages of design and production.

Failure Modes and Effects Analysis (FMEA) is a method to analyze potential reliability problems in the development cycle of the project, making it easier to take actions to overcome such issues, enhancing the reliability through design (Sharma, Kumar, & Kumar, 2008). FMEA is used to identify actions to mitigate the analyzed potential failure modes and their effect on the operations. Anticipating these failure modes, being the central step in the analysis, needs to be carried on extensively, in order to prepare a list of maximum potential failure modes.

The process for conducting a Failure Modes and Effects Analysis is summarized as follows:

1. Describe product or process
2. Define functions
3. Identify potential failure modes
4. Describe effects of failures
5. Determine causes
6. Direction methods or current controls

* Corresponding author. Tel.: +98 21 88021067.
*E-mail addresses:* vebrahimi@ut.ac.ir (V. Ebrahimipour), krezaie@ut.ac.ir (K. Rezaie), shokravi@ut.ac.ir (S. Shokravi).

0957-4174/$ - see front matter © 2009 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2009.06.033

7. Calculate risks
8. Take action
9. Assess results

It should be noted that a failure mode may be introduced after any change and updates are made to the product and process. Thus, FMEA might need to be reviewed (and updated) whenever a new product (or process) is being introduced, any changes are made to the operations or a change is made to the process design. By providing the engineers with a tool to assist in ensuring reliable and safe products and processes, FMEA grants certain benefits for project management. It emphasizes problem prevention and acts as a catalyst for teamwork and exchange of healthy ideas. It captures engineering knowledge and provides a focus for improved testing and development, eventually resulting in increased customer satisfaction.

## 2. OWL

Ontology-based method for knowledge representation offers a means for the reuse and sharing of knowledge unambiguously (Yang, Miao, Wu, & Zhou, 2008). OWL is a way of representing product configuration knowledge using semantic web technology; it is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans (García-Sánchez, Martínez-Béjar, Contreras, Fernández-Breis, & Castellanos-Nieves, 2006). OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called ontology. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to represent machine interpretable content on the Web. OWL is a revision of the DAML + OIL web ontology language incorporating lessons learned from the design and application of DAML + OIL (Berners-Lee, Hendler, & Lassila, 2001; Farquhar, Fikes, & Rice, 2002; Gangemi, Guarino, Masolo, Oltramari, & Schneider, 2002).

### 2.1. Why OWL?

The Semantic Web is a vision for the future of the Web, in which information is given explicit meaning, making it easier for machines to automatically process and integrate information available on the Web (Farquhar et al., 2002). The Semantic Web will build on XML's ability to define customized tagging schemes and RDF's flexible approach to representing data. The first level above RDF required for the Semantic Web is an ontology language what can formally describe the meaning of terminology used in Web documents (Yang, 2008). If machines are expected to perform useful reasoning tasks on these documents, the language must go beyond the basic semantics of RDF Schema. The OWL Use Cases and Requirements Document provides more details on ontologies, motivates the need for a Web Ontology Language in terms of six use cases, and formulates design goals, requirements and objectives for OWL.

OWL has been designed to meet this need for a Web Ontology Language. OWL is part of the growing stack of W3C recommendations related to the Semantic Web (Berners-Lee et al., 2001; Farquhar et al., 2002; Gangemi et al., 2002; W3C, 2004):

- XML provides a surface syntax for structured documents, but imposes no semantic constraints on the meaning of these documents.
- XML Schema is a language for restricting the structure of XML documents and also extends XML with datatypes.
- RDF is a datamodel for objects ("resources") and relations between them, provides a simple semantics for this datamodel, and these datamodels can be represented in XML syntax.
- RDF Schema is a vocabulary for describing properties and classes of RDF resources, with a semantics for generalization-hierarchies of such properties and classes.
- OWL adds more vocabulary for describing properties and classes: among others, relations between classes (e.g. disjoint ness), cardinality (e.g. "exactly one"), equality, richer typing of properties and characteristics of properties (e.g. symmetry), and enumerated classes.

### 2.2. The three sublanguages of OWL

OWL provides three increasingly expressive sublanguages designed for use by specific communities of implementers and users:

- OWL Lite supports those users primarily needing a classification hierarchy and simple constraints. For example, while it supports cardinality constraints, it only permits cardinality values of 0 or 1. It should be simpler to provide tool support for OWL Lite than its more-expressive relatives, and OWL Lite provides a quick migration path for thesauri and other taxonomies. Owl Lite also has a lower formal complexity than OWL DL, see the section on OWL Lite in the OWL Reference for further details.
- OWL DL supports those users who want the maximum expressiveness while retaining computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). OWL DL includes all OWL language constructs, but they can be used only under certain restrictions (for example, while a class may be a subclass of many classes, a class cannot be an instance of another class). OWL DL is so named due to its correspondence with description logics, a field of research that has studied the logics that form the formal foundation of OWL.
- OWL Full is meant for users who want maximum expressiveness and the syntactic freedom of RDF with no computational guarantees. For example, in OWL Full a class can be treated simultaneously as a collection of individuals and as an individual in its own right. OWL Full allows an ontology to augment the meaning of the pre-defined (RDF or OWL) vocabulary. It is unlikely that any reasoning software will be able to support complete reasoning for every feature of OWL Full.

Each of these sublanguages is an extension of its simpler predecessor, both in what can be legally expressed and in what can be validly concluded. The following set of relations hold. Their inverses do not:

- Every legal OWL Lite ontology is a legal OWL DL ontology.
- Every legal OWL DL ontology is a legal OWL Full ontology.
- Every valid OWL Lite conclusion is a valid OWL DL conclusion.
- Every valid OWL DL conclusion is a valid OWL Full conclusion.

Ontology developers adopting OWL should consider which sublanguage best suits their needs. The choice between OWL Lite and OWL DL depends on the extent to which users require the more-expressive constructs provided by OWL DL. The choice between OWL DL and OWL Full mainly depends on the extent to which users require the meta-modeling facilities of RDF Schema (e.g. defining classes of classes, or attaching properties to classes). When using OWL Full as compared to OWL DL, reasoning support is less predictable since complete OWL Full implementations do not currently exist.