Proceedings of DET2011
7th International Conference on Digital Enterprise Technology
Athens, Greece
28-30 September 2011

# KNOWLEDGE CAPITALIZATION INTO FAILURE MODE AND EFFECTS ANALYSIS

**Gabriela Candea**
Ropardo SRL
gabriela.candea@ropardo.ro

**Ciprian Candea**
Ropardo SRL
ciprian.candea@ropardo.ro

**Claudiu Zgripcea**
Ropardo SRL
claudiu.zgripcea@ropardo.ro

## ABSTRACT

To achieve high quality designs, processes, and services that meet or exceed industry standards, it is crucial to identify all potential failures within a system and work to minimize or prevent their occurrence or effects. This paper presents innovative usage of knowledge system in Failure Mode and Effects Analysis (FMEA) process. Knowledge system is built to serve multi-projects works that nowadays are in place in any manufacturing or services provider, and knowledge must be retained and reused not only at project level, but also at company level. Collaboration is assured through web-based GUI that supports multiple users' access at any time.

## KEYWORDS

Failure Mode and Effects Analysis (FMEA), Decision Support Systems (DSS), Collaborative Work, Case Based Reasoning (CBR), Knowledge Engineering, Knowledge Capitalization

## 1. INTRODUCTION

Preventing process and production problems before they occur is the purpose of Failure Mode and Effect Analysis (FMEA). Used in both the design and manufacturing processes, they substantially reduce costs by identifying product and process improvements early in the development process when changes are relatively easy and inexpensive to make. The result is a more robust process because the need for after-the-fact corrective action and late change crises are reduced and eliminated (McDermott, Mikulak and Beauregard, 2008). Product development is the result of a network-based collaborative process, because most of them require co-operation among geographically distributed experts with diverse competences (Mavrikios, Alexopoulos, Xanthakis, Pappas, Smparounis, Chryssolouris, 2011). The paper presents an innovative approach to FMEA that uses a knowledge system to capture and reuse content, a system developed by Ropardo S.R.L. for supporting the FMEA processes. The system is designed as a web collaborative tool that supports integrated multi project – multi team – multi language with knowledge repository system (*Experience Database*). At a higher level, this takes place inside an integrated system called *i*Portal (Cândea, Cândea, 2011), which is actually a software suite for different business related activities like project management, document management, decision support systems or other forms of collaboration.

In generic terms, knowledge is the internal state of an agent (in this case, the FMEA team member's experts) that has acquired and processed information from previous experience. An agent can be a human being, storing and processing information in his/her mind, or an abstract machine including devices to store and process information.

A body of formally represented knowledge is based on a conceptualization: the objects, concepts, and other entities that are assumed to exist in some area of interest and the relationships that hold among them (Genesereth & Nilsson, 1987). A conceptualization is an abstract, simplified view of the world that we wish to represent for our purpose. Every knowledge base, knowledge-based system, or knowledge-level agent is committed to some conceptualization, explicitly or implicitly. Innovation in our case resides in utilization of the Experience Database system as knowledge base for FMEA tool and as an active support (knowledge reuse) for FMEA team members in a multi-dimensional space: company – projects – teams.

Access to the entire system is provided via web interfaces with SSO (Single Sign-On) features so that attending the FMEA work sessions is done via web-browsers, not being restricted to localization.

Section 2 reviews briefly the existing FMEA software and processes how we found them in industry (automotive sector). Section 3 describes the architecture of the FMEA and Experience Database software systems, followed in Section 4 by deep Experience Database description, while the conclusions are presented in Section 5.

## 2. FMEA IN INDUSTRY

### 2.1 AN OVERVIEW OF BASIC CONCEPTS

Failure Mode and Effects Analysis (FMEA) and Failure Modes, Effects and Criticality Analysis (FMECA) are methodologies designed to identify potential failure modes for a product or process, to assess the risk associated with those failure modes, to rank the issues in terms of importance and to identify and carry out corrective actions to address the most serious concerns (Figure 1).

Although the purpose, terminology and other details can vary according to the type (e.g. Process FMEA - PFMEA, Design FMEA - DFMEA, System FMEA, Product FMEA, FMECA, etc.), the basic methodology is similar for all, one common factor has remained throughout the years—to resolve potential problems before they occur. For years, FMEA/FMECA has been an integral part of engineering designs. For the most part, it has been a necessary tool for industries such as the aerospace and automotive industries.

There are a number of published guidelines and standards for the requirements and recommended reporting format of Failure Mode and Effects Analyses. Some of the main published standards for this type of analysis include SAE J1739[1], AIAG FMEA-4 and MIL-STD-1629A. In addition, many industries and companies have developed their own procedures to meet the specific requirements of their products/processes.

FMEA/FMECA is a group activity (normally with 6-10 members), which may be performed in more than one sitting, if necessary. The process owner (or project manager) is normally the leader of the FMEA exercise; however, to obtain the best results, the process owner is expected to involve multi-disciplinary representatives from all affected activities. Team members should include subject matter experts and advisors as appropriate. Each Process Owner is also responsible for keeping the FMEA updated.

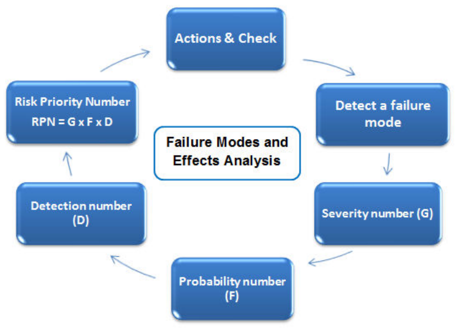


**Figure 1 - Main steps of FMEA**

### 2.3 THE FMEA/FMECA METHOD

'Failure modes' means the ways, or modes, in which something might fail. Failures are any errors or defects, especially ones that affect the customer, and can be potential or actual.

FMEAs are developed in very distinct phases where actions can be determined (Tague 2004). For FMEA, it is also required pre-work, in order to assure that the robustness and past history are included in your analysis. Bellow there are FMEA phases:

1. Identify the functions of your scope. Usually the scope will break into separate subsystems, items, parts, assemblies or process steps; identify the function of each.
2. For each function, identify all the ways failure could happen. These are potential failure modes.
3. For each failure mode (potential effects of failure), identify all the consequences on the system, related systems, process, related processes, product, service, customer or regulations.

[1] 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).

4. Determine how serious each effect is. This is the gravity rating, or G. Gravity is usually rated on a scale from 1 to 10, where 1 is insignificant and 10 is catastrophic. If a failure mode has more than one effect, write on the FMEA table only the highest gravity rating for that failure mode.
5. For each failure mode, determine all potential root causes. Use tools classified as cause analysis tool, as well as the best knowledge and experience of the team. List all possible causes for each failure mode on the FMEA form.
6. For each cause, determine the frequency rating, or F. This rating estimates the probability of failure occurring for that reason during the lifetime of your scope. Frequency is usually rated on a scale from 1 to 10, where 1 is extremely unlikely and 10 is inevitable.
7. For each cause, identify current process controls that might prevent the cause from happening, reduce the likelihood of occurring or detect failure after the cause has already happened (tests, procedures or mechanisms that keep failures from reaching the customer). For each control, determine the detection rating, or D. This rating estimates how well the controls can detect either the cause or its failure mode after they have happened but before the customer is affected. Detection is usually rated on a scale from 1 to 10, where 1 means the control is absolutely certain to detect the problem and 10 means the control is certain not to detect the problem (or no control exists).
8. (Optional for most industries) Is this failure mode associated with a critical characteristic? (Critical characteristics are measurements or indicators that reflect safety or compliance with government regulations and need special controls.) If so, a column labelled ‘Classification’ receives a Y or N to show whether special controls are necessary. Usually, critical characteristics have a severity of 9 or 10 and occurrence and detection ratings above 3.
9. Calculate the risk priority number, or RPN.

$$RPN = G \times F \times D$$

10. Identify recommended actions. These actions may consider design or process changes to lower severity or occurrence. They may be additional controls to improve detection. Also, who is responsible for the actions and target completion dates must be written.

As actions are completed, note results and the date on the FMEA form. Also, note new G, F or D ratings and new RPNs (Figure-2).

The RPN's are calculated after three possible action opportunities have occurred. Actions are not only determined based on RPN values. RPN threshold values do not play an important role in action development, only in action evaluation when completed.

| Item no. | Process Step/Input | Potential failure mode (deficiency) | Potential Failure Effects | G | Characteristic type | Potential Causes | F | Verification measures to prevent | Verification measures to detect | D | C (GxFxD) | Action plan | | The result of actions | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | | | Suggested remedies as required | Responsible; Term | Data verification / measures taken | G | F | D | C |
| 1 | unload cast | hits, dirty cast | cast in quarantine area, delay in fabrication, FIFO not respected | 4 | M | broken box, not proper storage | 2 | I-063.02 unload the supplied material and cast from the transport trucks | the supplied product quality acceptance procedure PL 075.30 | 6 | 48 | | | | | | | |

**Figure 2 FMEA example**

## 2.4 THE FMEA CHALLENGE

The management of a company’s knowledge is made difficult mainly by two issues: relevant knowledge may often not be found in a clear form like databases, but in documents (project statements, client requirements and QM handbooks) and the access to knowledge is overloaded by the problem that different actors use different terms to refer to the same topic.

Today, FMEA is in widespread use by a multitude of industries, many of which have begun imposing FMEA standards. The purpose of the FMEA is to take actions to eliminate or reduce failures, starting with the highest-priority ones (Tague, 2004). Nevertheless, the effort to develop an FMEA is mainly considered as highly or very highly due to the number of involved persons (Stock & Stone & Tumer, 2003). In addition, the advantages that result out of failure prevention cannot be perceived immediately. To shorten the process of FMEA development and earning results, the knowledge included in already developed FMEA has to be reused.

One of the most important keys of the FMEA is the capitalization of knowledge. When workers leave a company and take with them valuable job-related information, managers and co-workers are left to manage the new employees, disregarding their own responsibilities. Another weakness of FMEA is the experience of the FMEA team members, thus the FMEA is only as good as the members of the FMEA team.

The FMEA knowledge reuse suffers from a major shortcoming mentioned by Wirth et al., 1996: the FMEA-related information is acquired in natural language. The analyses are hardly reusable because the systematized components, functions and failure modes are not made explicit. The meaning depends on the interpretation of the team/ a team member who performs the FMEA and can fluctuate when another team reuses this FMEA, or even if the same team tries to reuse it on a later occasion. Caused by the lack of reusability the FMEA is regularly built from scratch without making use of older FMEAs.

Although one person typically is responsible for coordinating the FMEA process, all FMEAs are team based. The scope for a FMEA team is to bring a range of perspectives and experiences in the project. Because each FMEA is unique in dealing with different aspects of the product or process, FMEA teams are formed and dispersed when is needed[2]. Another limitation of the FMEA methodology is sets by the unavailability (de-located team, overlap of membership between the teams) of team members to attend at FMEA meeting.

## 3. SOFTWARE SYSTEMS

In this chapter the software system is presented from concept to architecture. Our system is composed by two major subsystems – PEA and Experience Database.

PEA (Process and Effect Analysis) is respecting all FMEA requirements and processes of work; it is a web-based software that allows team collaborative work on FMEA.

Second major system is Experience Database (Knowledge Repository System) that provides knowledge capitalization for our system. In current implementation Experience Database uses for capitalization of knowledge a case base reasoning (CBR) approach.

### 3.1 PEA – PROCESS AND EFFECT ANALYSIS

The purpose of FMEA software by Ropardo – (named PEA) is preventing process and production problems before they occur. It is used both in design and manufacturing processes and it substantially reduces costs by identifying product and process improvements early in the development process when changes are relatively inexpensive to implement. Process and Effect Analysis (based on FMEA) processes are based on worksheets that contain important information about the system, such as the revision date or the names of the components. On these worksheets all the items or functions of the subject should be listed in a logical manner, based on the block diagram. For each item or function, the possible failure modes, effects and causes are listed and each of them are graded for their severity/gravity (G), frequency of occurrence (F), and detection rating (D). Afterwards, the Risk Priority Number (RPN) is calculated by multiplying S, F and D. Once this is done it is easy to determine the areas of greatest concern. This has to consider the entire process and/or design and the items that have the highest RPN should be given the highest priority for corrective actions. After these values are allocated, recommended actions with targets, responsibility and dates of implementation are noted on the worksheets which actually consist in the output of this software.

### 3.2 EXPERIENCE DATABASE

The experience database aims to provide an easy to use component by the knowledge engineer and by other software modules.

A knowledge management system faces on few major challenges: 1) Acquisition – The main target here is to get hold of the information that is around, and turn it into knowledge by making it usable. This might involve, making tacit knowledge explicit, identifying gaps in the knowledge already held, acquiring and integrating knowledge from multiple sources. 2) Modelling – Knowledge model structures must be able to represent knowledge so that it can be used for problem-solving. One important knowledge modelling idea is that of ontologies, which are specifications of the generic concepts, attributes, relations and axioms of a knowledge base or domain. Ontologies can act as placeholders and organizing structures for acquired knowledge, while also providing a format for understanding how knowledge will be used.

3) Retrieval – When a knowledge repository gets very large, finding a particular piece of knowledge can become very difficult 4) Reuse – On problem in using knowledge management systems is that often knowledge databases are rebuilt for each end user. 5) Publishing – can be described as follows: knowledge, in the right form, in the right place, to the right person, at the right time. 6) Maintenance –

[2] http://www.fmeainfocentre.com/, FMEA Info Centre

It may involve the regular updating of content as content changes. In addition, it may also involve a deeper analysis of the knowledge content.

The Experience Database component whose main architecture is described in Figure-3 is using Case-based Reasoning as computational engine. Actual design of Experience Database allows defining and storing different types of structures for knowledge representation. These structures can be defined using ontologies editor that allow you to keep an organized and easy way to access and view the database. In our case we are using Protégé as ontologies editor and defined ontology is stored using Protégé internal storage. We built a mapping tool that allows exporting certain structure from ontology to the Experience Database as showed in Figure-3. Once that structure exported to CBR engine, we will operate two atomic structures, one original that is defined inside of ontology and the other one, inside CBR system. In this way ontology can evolve and new case structures can be created any time, on the other side – CBR – once that case is created and populated with data, this structure is fixed and structure can be modified only manual – no automatic update process.

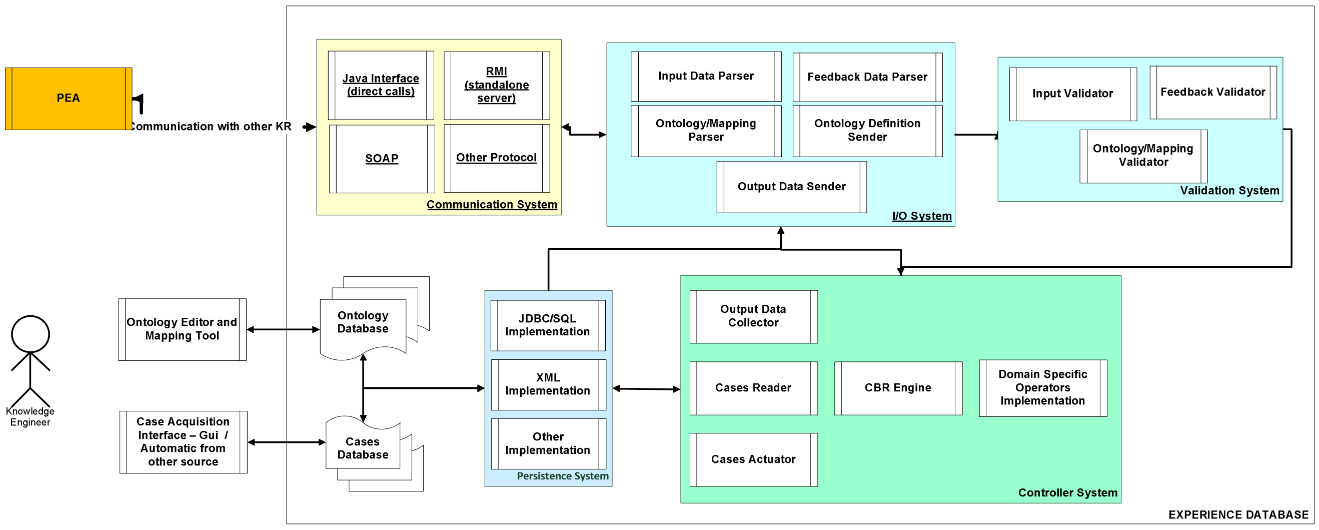


**Figure 3 – Experience Database architecture**

The communication system assures the independence of the module core processing model from the communication methods. The default implementation it is the direct Java calls: the client will get a communication object, which exposes the methods through a Java interface. The methods are invoked by direct calling, all the data types being passed without transformation (into/from XML or similar); other methods are exposed by Experience Database and can be used as well.

One important sub-system of Experience Database is the input/output (I/O), validation that is responsible for the translation of the data from external sources into native data types, which can be used by the controller.

The I/O system is split into two subsystems, one for input and one for output:

- The input subsystem achieves the translation of information from generic formats (XML structures) into Java formats (POJO – Plain Old Java Objects). This is done by validating and parsing the XML input into the corresponding POJO. The validation is done against the XSD and it is different from the validation done within the validation system – it consists only in checking if the syntax of XML is correct.
- The output subsystem generates the XML answers from the Java objects (it is mainly a serialization of Java objects into the corresponding XML representation, but additional transformations may apply).

The validation system checks the incoming data for inconsistencies and rejects the wrong ones.

Input Data Parser – it is responsible with the parsing of the input (request) information. The input data requests are for similar cases (a search over the stored cases using some filtering parameters) or request for a single case (identified by its ID).

Feedback Data Parser – it is responsible for the parsing of the feedback data. The feedback consists in changes to a stored case (different solution, etc.).

Ontology/Mapping Parser – it is responsible for the parsing of the domain/case ontology and with the parsing of the mapping information that will be used by the controller to solve the problem. The mapping information is domain dependent and will be defined by the knowledge engineer. The default implementation will provide some default mappings, but other will be needed to be defined.

Ontology Definition Sender – it is responsible for the formatting of the ontology definition from the internal format into the XML file. The sender is invoked by the controller upon a corresponding request, which is received. Moreover, the ontology is fetched from the database (please see relevant sequence charts).

Output Data Sender – it is responsible for the formatting of the retrieved cases/answers into the correct XML structures.

Input Validator – needs to validate the parsed input data for inconsistencies.

Feedback Validator – needs to validate the feedback information for inconsistencies.

Ontology/Mapping Validator – needs to validate the ontology and domain mapping information for inconsistencies.

All the validation is done in order to lighten the controller processing (the controller receives only good information; the wrong input will be filtered before).

The standard invocation process starts a search in experience databases – search that is done on criteria of similarity functions (detailed description on next chapter) that are defined for each data structure. The search is done separately for each data structure defined in separate spaces (case space) in its database for a better case management. To start a new search, an XML containing the case pattern data is sent to Experience Database and, as response, an XML with the best 'n' cases is returned. For the feedback phase PEA will send an XML with feedback data for the specific case pattern based on algorithms that Experience Database "learns".

## 5. CBR ON EXPERIENCE DATABASE

In our implementation of Experience Database, a Case Base Reasoning engine is the core computational engine that solves problems by adapting solutions to older ones.

A CBR system involves reasoning from prior examples, memorizing previous problems and associated solutions and solving new problems by referencing to that knowledge. (Sankar K. Pal, Simon C.K. Shiu, 2004).

The problem-solving life cycle in our CBR system consists essentially of the following four parts as in Figure 4.

- Retrieving similar previously experienced cases (e.g., problem–solution–outcome triples) whose problem is judged similar
- Reusing the cases by copying or integrating the solutions from the cases retrieved
- Revising or adapting the solution(s) retrieved in an attempt to solve the new problem
- Retaining the new solution once it has been confirmed or validated

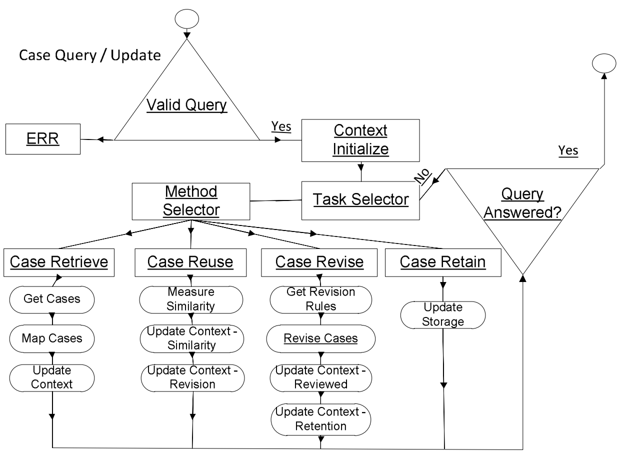


**Figure 4 – CBR internal design**

### 5.1 CASE STRUCTURE

We start from a general case structure definition for our implementation for FMEA process and contain the following information.

- ID. A unique identification number of the case base.
- Description. A brief description for the case.
- Meta-data. The case meta-data is maintained for each case.
- Creator. Name of person/ organization / project that created the case.
- Creation date/time. Date and time the case was initially saved in the case base.
- Number of times accessed. Count of the number of times the case has been retrieved from the case base by a client.
- Date/time of last access. The date/time of the last time the case was retrieved.
- Features. A list of case features. A case feature is synonymous with a case index.
- Data or Sub-cases. This is also commonly referred to as the case solution. The case data (solution) contains the information that is returned to the client during case retrieval. If a case has child cases no data is associated with the parent case. For these aggregate cases, a list of child cases is maintained.

Starting from this general definition we defined for FMEA a specific case schema and which respects our scope for knowledge capitalization on multi project and multi user usage.

We start from specific representation of FMEA domain and we built the case structure where we can find information about process/product, as well as effects and measures that must be taken for each case.

For example, we can consider next case structure.

PEA's Case

PN – Project Name

P – Product / Process

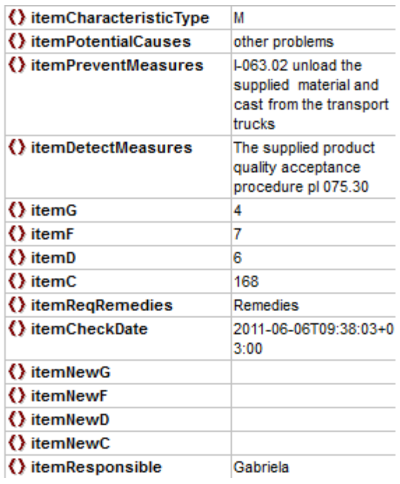
PE – Effect

PC – Potential Cause

The solution

NG – New Gravity

R – Remedy



| | |
|---|---|
| {} itemCharacteristicType | M |
| {} itemPotentialCauses | other problems |
| {} itemPreventMeasures | I-063.02 unload the supplied material and cast from the transport trucks |
| {} itemDetectMeasures | The supplied product quality acceptance procedure pl 075.30 |
| {} itemG | 4 |
| {} itemF | 7 |
| {} itemD | 6 |
| {} itemC | 168 |
| {} itemReqRemedies | Remedies |
| {} itemCheckDate | 2011-06-06T09:38:03+03:00 |
| {} itemNewG | |
| {} itemNewF | |
| {} itemNewD | |
| {} itemNewC | |
| {} itemResponsible | Gabriela |

**Figure 5 – Part of PEA case structure**

## 5.2 SIMILARITY FUNCTION

Case retrieval is the process of finding, within a case base, those cases that are the closest to the current case. For an effective case retrieval, we start from selection criteria – in our case, a partial case structure completed on the GUI by the user – that determines how to compute a case to make it appropriate for retrieval. Starting from selection criteria the closed case is searched through the cases stored in the database. The most commonly investigated retrieval techniques, are the k-nearest neighbours (k-NN), decision trees, and their derivatives. These techniques involve developing a similarity metric that allows the closest (i.e., similarity) cases to be measured.

For example, if we are looking to find similar cases to query case qc = (PN, P, PE, PC) case retrieval is the process of finding what case is the closest (cc) one to qc.

For each case from the database is calculated the degree of similarity ➔ equation-1, between qc and cci; i=1 to n; where n is the total number of cases in the database.

$$SM(qc, cc_i) = \frac{common}{common + diffrent} \qquad (1)$$

Where "common" represent the number of feature whose value is the same between qc and cci, and "different" represents the number of features whose value is different between qc and cci.

In current implementation we are implementing a similarity function that is based on the Euclidian weighted distance ➔ equation-2. The distance is calculated as the square root of the sum of the squares of the arithmetical differences between the corresponding coordinates of two objects (Sankar K. Pal, Simon C.K. Shiu, 2004).

$$d_{pq}^{w} = \sqrt{\sum_{j=1}^{n} w_j^2 \rho_j^2 (e_{pj}, e_{qj})} \qquad (2)$$

Where w is the weight of the associated j the feature that indicates the importance of that feature $w_j \in [0,1]$.

For distance measure computation, we used next formulas.

- $\rho_j(a,b) = |a-b|$ if a and b are real numbers
- $\rho_j(A,B) = \max_{a \in A, b \in B} |a-b|$ if A and B are intervals
- $\rho_j(a,b) = \begin{cases} 1 & if \quad a \neq b \\ 0 & if \quad a = b \end{cases}$ if a and b are symbols

## 6. CONCLUSIONS

In current implementation, we are proposing a method to mobilize the professional knowledge of the professionals involved into FMEA process. Nowadays, in manufacturing sector, decisions concerning processes and products must be anticipated by integrating the professional knowledge and know-how of experts from early stages to motorization and correction. Different aspects where investigated from artificial intelligence (Barthe` s, J.P., 1996), Case-Base Reasoning (Haque, B.U. et al., 2000) and knowledge management of knowledge capitalization.

We are proposing an innovative method that allows knowledge capitalization for FMEA process. Moreover, we design and built the software system that, on the one hand offers a new approach for FMEA standard – collaborative on multi user, multi project using web GUI – PEA software; and on the other one we put together the Experience Database with the FMEA specific knowledge capitalization.

As a core computational engine for Experience Database we used Case Base Reasoning engine and for similarity function, we implemented Euclidian weighted distance.

The software system presented in this article is going to be lunched in production to the biggest automotive spare parts supplier from Romania, starting with Q4 2011.

As future work, we must investigate different similarity functions and we are looking to implement and evaluate fuzzy approach.

Other task that must be accomplished is the maintenance of the CBR system whose current configuration may become sub-optimal over time, and therefore is critical to have the ability to optimize the configuration.

## 7. ACKNOWLEDGMENTS

The research reported in this paper has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement No: NMP2 2010-228595, Virtual Factory Framework (VFF).

## REFERENCES

Barthe` s, J.P. (1996). ISMICK and Knowledge

Cândea, C., Cândea, G. (2011). "iPortal and Factory Hub", Digital Factory for Human-oriented Production Systems: The Integration of International Research Projects, L. Canetta, C. Redaelli, M. Flores (eds), 1st printing, Springer, pp. 271-282

Cândea, C., Georgescu, A.V., Cândea, G. (2009). iPortal – Management Framework for Mobile Business. Proceedings of the International Conference on Manufacturing Science and Education MSE.

Filip, F.G. (2007). Decision Support Systems. 2nd edition revised, pp 353 – 359. Ed. Tehnica, Bucuresti.

Georgescu, A.V., Cândea, C., Constantin, Z.B. (2007). iGDSS – Software Framework for Group Decision Support Systems. In Procedings of The Good, The Bad and The Unexpected.

Goddard, P.L. (2000). Software FMEA techniques. Reliability and Maintainability Symposium. pp 118-123.

Haque, B.U. et al. (2000). Towards the Application of Case Reasoning to Decision-making in Concurrent Product Development (Concurrent Engineering), School of Mechanical, Manufacturing Engineering and Management, pp. 101–112, University of Nottingham, Nottingham, NG7 2RD, UK.

Huang, G.Q., Shi, J., Mak K.L. (2000). Failure Mode and Effect Analysis (FMEA) Over the WWW. The International Journal of Advanced Manufacturing Technology, Vol. 16 (8), pp. 603-608.

Mavrikios D., Alexopoulos K., Xanthakis V., Pappas M., Smparounis K., Chryssolouris G. (2011). "A Web-based Platform for Collaborative Product Design, Review and Evaluation", Digital Factory for Human-oriented Production Systems: The Integration of International Research Projects, L. Canetta, C. Redaelli, M. Flores (eds), 1st printing, Springer, pp. 35-55

McDermott, R.E., Mikulak, R.J., Beauregard, M.R. (2008). The basics of FMEA. Productivity Press, USA

Puente, J., Pino, R., Priore, P., Fuente, D. (2002). A decision support system for applying failure mode and effects analysis. International Journal of Quality & Reliability Management. Vol 19 (2), pp 137-150.

Sankar K. Pal, Simon C.K. Shiu, (2004). Foundation of Soft Case-Based Reasoning

Schreinemakers, J.F. Knowledge Management: Organization, Competence and Methodology, Proceeding of ISMICK'96, pp. 9–13, 21–22 Oct, Rotterdam, Wurzburg, Ergon Verlag.

Stamatis, D.H. (2003). Failure mode and effect analysis: FMEA from theory to execution, William A. Tony, USA

Tague, N.R. (2004). The Quality Toolbox, Edition, ASQ Quality Press, pp 236-240.

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

Automotive Industry Action Group, "Implementing the Failure Mode & Effects Analysis (FMEA) Reference Manual", Retrieved: 8 August 2011, http://www.aiag.org/scriptcontent/index.cfm,

FMEA Information Centre Team, Retrieved: 8 August 2011, FMEA Info Centre, http://www.fmeainfocentre.com/

Automotive Quality and Process Improvement Committee, "Potential Failure Mode and Effects Analysis in Design (Design FMEA), Potential Failure Mode and Effects Analysis in Manufacturing and Assembly Processes", Society of Automobile Engineers, Retrieved: 10 August 2011, http://standards.sae.org/j1739_200901