Knowledge-Based Systems, Volume 26, February 2012, Pages 48–60

Knowledge-Based Environment to support product design validation

A. Al-Ashaab[1], M. Molyneaux[2], B. Brunner[1], E. Martínez[1], F. Moliner[1], V. Santamaría[1], D. Tanjore[1], P. Ewers[2], G. Knight[2], A. Doultsinou[1]

[1]*Manufacturing Department, School of Applied Sciences, Cranfield University, MK43 0AL, United Kingdom*

[2]*Visteon Engineering Services Ltd, Chelmsford, Essex, CM2 5LB, United Kingdom*

Abstract

A Knowledge Based Environment Knowledge-Based Environment to Support Product Design Validation of Refresh Projects (projects with minor changes in respect to previous ones) has been developed on Teamcenter Platform (PLM software). Refresh Projects are around 60% of the total number of projects in the sponsoring company. This implementation will avoid repeating unnecessary and costly physical product tests saving time and costs by at least 10% for these Refresh Projects. The KBEnv has been developed by capturing the relation between design change of refresh projects with their required physical tests and the historical data of previous projects. The KBEnv framework has been implemented on Teamcenter Platform after standardising the documents and maintaining the traceability of decision making process. The KBEnv Framework Implementation is estimated to reduce the overall communication time by 52%, reduce the deviation in output by 45% and enhance confidence in decision making by 20 %.

Keywords—Knowledge Based Environment, PLM, Teamcenter, Product design validation.

1. Introduction

The recession period has been especially challenging for the automotive industry and its suppliers. They are under constant pressure to deliver extra functionalities in their products and at the same time, reduce costs and the lead time for the design. Advanced IT solutions can be very beneficial to increase the ability to innovate, get products to market faster, reduce errors and the final cost.

In a typical product development process, there is a need to have an independent design validation. This is usually done by an independent team or department through performing climatic and EMC (Electromagnetic Compatibility) tests. In reality, many new product development projects are based on the previous designs with either major or minor modifications. The latter are called “Refresh Projects”. It is also a fact that there is a little variation in the test results of refresh projects compared to the previous projects. As such, it is highlighted that not all the tests performed are necessary. In addition, it is important to follow a standard approach to develop a Design Validation Plan (DVP) for these refresh projects.

This paper is presenting the work of developing a Knowledge Based Environment (KBEnv) on the PLM platform Teamcenter (TcSE) based on historical data and the observation of the design changes. This work has been done for a first tier automotive parts supplier company. The intention is to provide a rich environment based on logic and facts to avoid repeating physical tests on these refresh projects in order to reduce the development cost and time. This research has developed a KBEnv that would make a drastic improvement and optimise not only the product validation but also streamline the overall process of design validation planning; this has been called by the authors KBE-ProVal (*K*nowledge-*B*ased *E*nvironment to Support *Pro*duct Design *Val*idation).

In this paper, the research methodology is firstly presented followed by the theoretical knowledge foundation. Then, the stages adopted in the research methodology: Refresh Project Analysis, Development of KBEnv Architecture, Implementation of KBEnv Framework, Case Study Validation are explained in detail. Enhancement indicators have been qualitatively and quantitatively defined to highlight the progress achieved by the implementation of KBEnv in product design validation.

2. Research methodology

The research methodology developed to achieve the objectives of the project is illustrated in Figure 1. It consists of five stages where each stage entails several tasks.

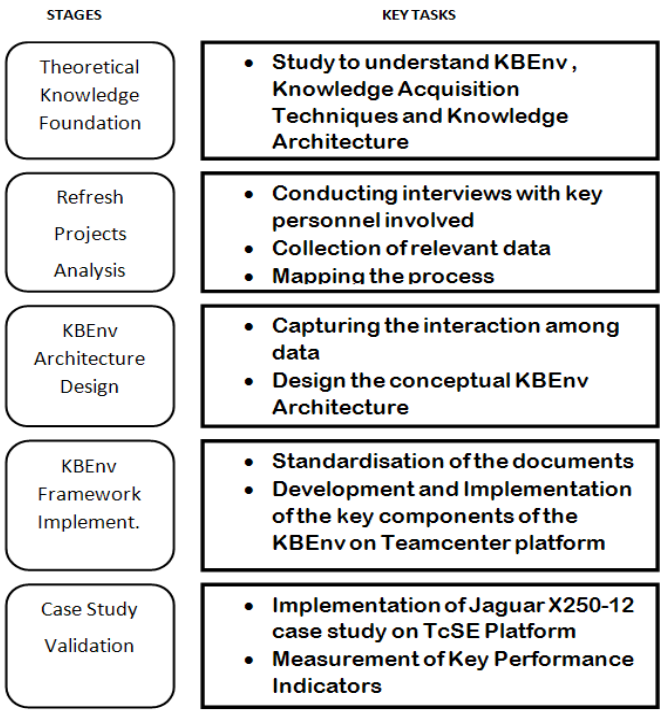


Figure 1: Research Methodology

The different stages and the activities performed are as follows:

1. Theoretical Knowledge Foundation: This is based on literature review and analysis to support the understanding of the key concepts of the project. This is done through the following tasks:
   1.1. A review of the Knowledge Life Cycle (KLC) approaches.
   1.2. Understanding the concept and role of KLC in KBEnv.
2. Refresh Projects Analysis: This is to develop the AS-IS model of the product assurance of the refresh projects. This is done through the following tasks:
   2.1. Conducting semi-structured interviews with the key personnel involved in product development.
   2.2. Collection of the relevant data such as previous projects test failures, design changes, etc.

2.3. Mapping the process using the IDEF0 technique to identify opportunities of improvement through the development of knowledge environment based on the captured data and their interactions.

3. KBE-ProVal Architecture Design: This is to evaluate the key components needed for the implementation and the interaction between them. This is done through the following tasks:
   3.1. Identifying the main elements and their interactions that form the Management of Change, which is the core element of the KBE-ProVal.
   3.2. Designing the conceptual KBE-ProVal architecture framework.
   3.3. Planning the implementation of the KBE-ProVal architecture framework on the Teamcenter platform.
4. KBE-ProVal Implementation on TcSE: This is to implement the key components of the KBE-ProVal architecture on the Teamcenter platform. This is done through the following tasks:
   4.1. Studying the features and capabilities of TcSE.
   4.2. Standardising the key documents in KBE-ProVal Architecture.
   4.3. Implementing the Management of Change framework and the key components of the KBEnv architecture on TcSE maintaining the traceability of decision making.
5. Case Study Validation: This is to validate the Management of Change framework implemented on the Teamcenter framework. This is done through the following tasks:
   5.1. Implementing a case study on the developed KBE-ProVal on TcSE, following a step by step process.
   5.2. Identifying and capturing the key enhancement indicators.

The approach is sequential, where the outputs of one stage were the inputs to the other stage. The development in every stage has been closely monitored and the outputs of every stage have been

validated before proceeding to the next stage. The following sections present the paper according to the key stages presented in the research methodology.

3. Adopted Knowledge Life Cycle Approach

According to Dalkir (2005), one of the major decisions in designing a KBEnv is to initiate a Knowledge Lifecycle (KLC). A KLC transforms information to knowledge, a valuable asset that can be used in decision making. An Integrated overall KLC consists of three main stages:

- Knowledge capture and/or creation
- Knowledge sharing and dissemination
- Knowledge acquisition and application

There are many models of approaches for KLC. However, there are six major established approaches to achieve KLC. These are shown in Table 1.

Al-Ashaab& Rodriguez (2004)	**Meyer & Zack (1996)**	**Buckowitz & Williams (2003)**	**McElory (1999)**	**MOKA (2001)**	**Wiig(1993)**
Identify	Acquisition	Get	Learning	Identify	Creation
Capture	Refinement	Use	Knowledge Claim Validation	Justify	Sourcing
Represent	Store/retrieve	Learn	Info Acquisition	Capture	Compilation
Implementation	Distribution	Contribute	Knowledge Validation	Formalise	Transformation
Use	Presentation	Assess	Knowledge Integration	Package	Dissemination
Create		Build/Sustain		Activate	Application
Maintain and up-grade		Divest			Value Realisation

Table 1: Summary of Knowledge Life Cycles approaches.

From these approaches the KLC of Al-Ashaab and Rodriguez (2004) was chosen, since the processes had to be well identified and moreover a lot of focus had to be laid on capturing, standardising and representing the data in the right format before proceeding to Implementation. According to Al-Ashaab and Rodriguez KLC (2004) the steps have been explained as follows:

1. Identify and gather knowledge from different sources.
2. Capture/acquire and standardise the knowledge and constraints.
3. Represent the knowledge formally.
4. Implementation in a knowledge based system
5. Use of the knowledge to support decision taken.
6. Create new knowledge through experiments
7. Maintain and upgrade knowledge.

Knowledge Architecture is more focused on designing "space" where knowledge can be created, captured and shared. Thus, the actual content is not of that prime importance as the life cycle is, i.e. how and when the knowledge is created and how it reaches the right people at the right time.

4. Refresh projects validation process

The product assurance (PA) activity consists of establishing a Design Validation Plan (DVP) which is the list of tests that need to be performed to validate a product. Traditionally, the DVP was produced manually, while with the use of the tool the KBEnv will produce the results automatically in the same format as before. This section first presents the approach adopted to capture the product assurance process for refresh projects, and then explains the AS-IS model captured. The analysis of the AS-IS model will help to identify the key components of the scenario, as well as opportunities of improvement.

4.1. Approach to capture refresh project design validation process

The product design validation process is not isolated from the rest of the product lifecycle activities. The formation and collection of knowledge is deeply involved with all product lifecycle activities, including

its design, manufacturing, validation and distribution processes. Thus, an effective management, integration and sharing of the knowledge involved in all these activities, is needed (Chen et al., 2009).

The first step was to understand the process, identify and collect the knowledge required from different sources. The understanding of the authors is shown in Figure 2 by a pictorial representation.

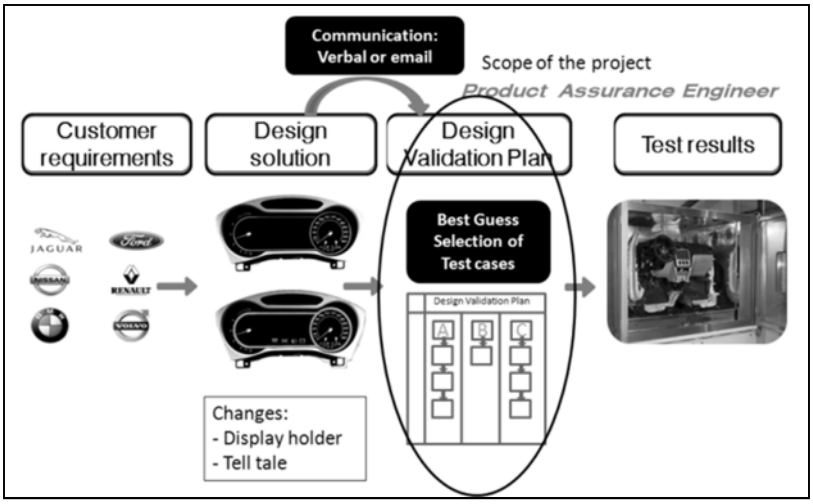


Figure 2: Product Assurance process for refresh projects.

Figure 2 illustrates that customer requirements are delivered to the company and the design team provides a product design solution according to these requirements. The design team consists of electrical, mechanical and software engineers. The Product Design Team Leader (PDTL) is the engineer responsible for gathering the design solution and giving it to the PA engineer. The PA engineer must develop the list of tests to be performed to validate the product design present in the Design Validation Plan. The circled area in

Figure 2 shows the focus of this study.

The next step was to capture the knowledge of the product development process from different sources presented in the scenario shown in

Figure 2. Firstly, several engineers involved in the present scenario were identified. Basic research was conducted regarding the role of these individuals and the semi-structured interview technique was selected to acquire the information needed from them. The aim and the background of the interviews were carefully prepared. In the next paragraph, the main ideas obtained throughout the interviews are presented.

After interviewing electrical, mechanical and software engineers it was realised that each of these groups worked in a different way from the customer requirements to the product design solution. Hence, it was decided to limit the scope of this work to the mechanical part of a design.

When the customer requirements are delivered to the company, a requirements team is in charge of generating a document following the company standard. This document is called Specifications Requirements Document (SRD). However, the design team uses directly the customer requirements to design a solution due to the time taken to generate this document (up to three months). A mechanical engineer interviewed underlined that the customer requirements always led to a design solution. This statement determined that the input of the future framework would be the design solution rather than the customer requirements. If this design solution is compared to a previous project, a mechanical engineer is able to highlight the changes easily. After summarising all the interviews, a brainstorming session helped to put all the ideas together in order to map the whole process.

4.2. AS-IS model of product assurance for refresh projects

Modelling the product assurance process is essential for developing the KBE-ProVal architecture at a later stage. To represent the product assurance process for refresh projects, the IDEF0 technique has been selected due to the fact that it can be easily understood by the technical developers and also by the

industrial end-users. The IDEF0 technique starts with a general overview of the whole process and it is decomposed to several lower levels (Shen et al., 2004).

In the IDEF0 model shown in

Figure 3, all the activities that take place during the PA process for refresh projects are presented in a logical order. In the next lines, all the knowledge identified and acquired throughout the process will be detailed.

The PA process for refresh projects is different than the process for a new project. The engineering changes are defined by the customer requirements, which can be received by e-mail, phone or personal meetings without a standard format.

The first activity performed in the process is “Produce SRD” (A1). This activity consists of generating the Specifications Requirements Document (SRD) from the customer requirements.

The second activity is “Design solution” (A2). The mechanical engineers and the PDTL are involved in this task. The mechanical engineers receive the customer specifications from the Requirements Team or directly from the customer and design a solution. The PDTL reviews and shares this design solution with the PA by e-mail or verbal communication.

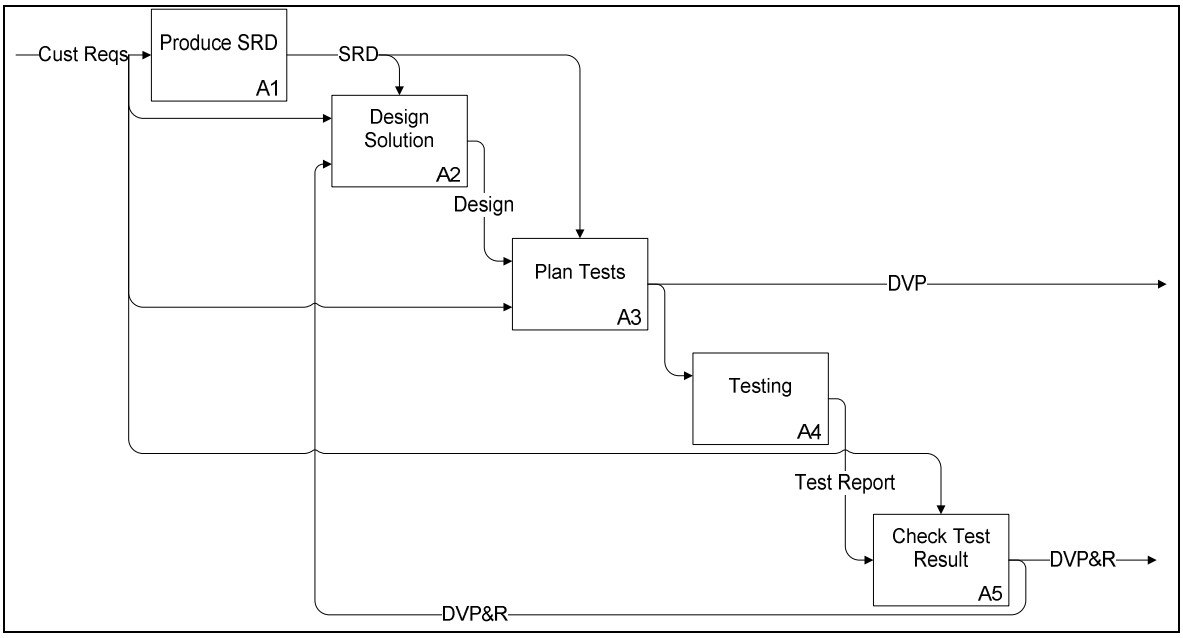


Figure 3: AS-IS model of product assurance for refresh projects.

According to the design changes proposed for the new product and his/her personal knowledge; the PA will develop a first design validation plan to be validated by the customer. This is the "Plan Tests" activity (A3). After the customer's agreement, the DVP is sent to the test facilities. The activity "Perform tests" (A4) takes place in the test facilities. Once the planned tests have been performed, the test results are sent to the PA to be checked. This is the last activity of the process, called "Check Test Results" (A5). This activity generates as an output the DVP&R, which is locally stored by the PA and sent to the customer. In the case of a test failure, the result is shared with the design team to fix the problem or redesign the product if necessary. After mapping the actual process, the next step consists of designing the conceptual KBE-ProVal architecture.

5. KBE-ProVal architecture design

The conceptual KBE-ProVal architecture must support the decision making process of the PA engineer for refresh projects allowing the traceability between the design solution and the design validation plan. Figure 4 illustrates this KBE-ProVal architecture. The main elements that comprise this architecture are: the theoretical knowledge, the definition of change, the historical test data and the management of

change. The reason behind that is the whole knowledge-based environment is based on the historical test data standardisation, the product design change and the risk assessment through theoretical knowledge. The interactions among them are managed by the management of change (MoC).

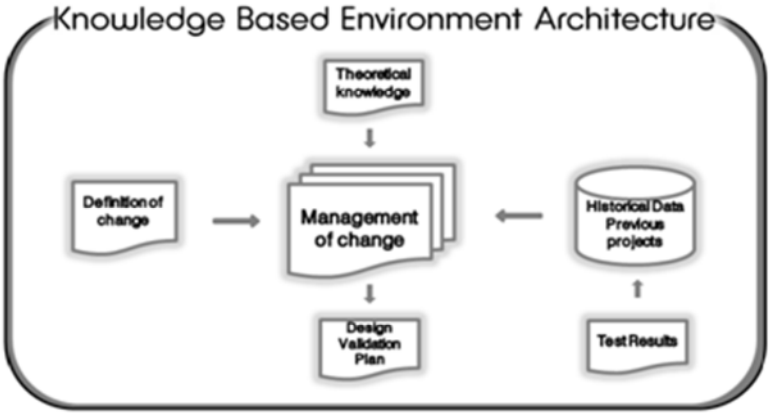


Figure 4: Conceptual KBE-ProVal Architecture

The KBE-ProVal to be designed must provide several functionalities. According to the actual design validation process, the aim is to develop architecture able to support the retrieval of previous projects test results data, to evaluate the risk of performing a test according to the previous projects failures and to help highlighting the engineering design changes and their effects on the product's subsystems. The key components of the KBE-ProVal architecture as shown in Figure 4 are the 'Definition of Change' (DoC), the 'Theoretical Knowledge' (TK) and the 'Historical test Data' (HTD) of previous projects as presented in the following sub-sections.

5.1 Definition of Change

The product design changes that define a refresh project need to be highlighted by the design engineers. In the case of the mechanical engineers, they analyse the different components of the product that have changed. The process of highlighting the changes for a refresh project is called "Definition of Change" (DoC). The change categories must be common throughout the family of products, and must also

adequately define how a component has changed with respect to its performance in any given environment. An environment can be simulated for example hot temperature testing or chemical resistance testing. The change categories are critical to the functionality of the KBE-ProVal because they must at some point be linked to the other elements of the system in order for them all to contribute towards the same overall output which is to accurately assign risk to each individual test case (as a result of engineering change). The following are the seven product design change categories specifically related to complex electronic assembly (also shown in Figure 9):

1. Electrical Performance - an indicator of electrical properties (e.g. Microprocessor threshold voltage changed to 5 volts)
2. Clearance - an indicator of proximity of one component to another (e.g connector moved closer to plastic cover
3. Retention - an indicator of component-to-component mating integrity (e.g. LCD display clipped to PCB)
4. Sealing - an indicator of ingress performance (e.g. rubber seals around back cover)
5. Mass or Stiffness - an indicator of mass or stiffness (e.g. heatsink mass increased from 120 grams to 180 grams)
6. Material - an indicator of material properties (e.g. connector with zinc plating over brass base material)
7. Thermal Management - an indicator of temperature (e.g. LCD runs 4 degrees hotter than previous design).

The DoC is essentially a matrix of components versus the above seven product design change categories. Only the significant components of a product are defined which results in approximately 20 component categories for the product. Each of the 7 change categories apply to each of those 20 component categories, resulting in a maximum of 140 data fields to fully define the changes on a refresh

project. In reality most of the data fields are left blank to denote that there have been no changes made in that area (see Figure 10). The content of each data field is a single number on a scale of 1 to 3 to define the severity of the change. For instance, a mass change of less than 1% could result in a rating of 1 and a mass change of more than 50% could result in a rating of 3, against any particular component (Please refer to Figure 10 for a screenshot of the model which could provide context for the above explanation).

Due to these changes, different product attributes can be affected. An example of these attributes could be the electrical performance, the material or the thermal management. The categorisation of these attributes has been validated by a senior mechanical engineer.

To assist the decision making process and select whether a validation test is applicable or not, it is necessary to understand these effects. A justification needs to be made as well as risk evaluation.

5.2 Theoretical Knowledge

The PA Engineers do not follow a standard approach to develop the DVP. The process to generate this DVP is the following: The design solutions for the refresh projects are communicated verbally or by email to the PA engineer. To achieve his/her task, the PA engineer performs an individual evaluation of the risks associated to each test based on personal tacit knowledge. Nevertheless, for the same design solution, another PA engineer who evaluates the risks may do it differently and the resulting DVP could be different.

To address the problem of sharing and centralising the tacit knowledge of PA engineers, the ‘Theoretical Knowledge’ to be designed became a key component of the architecture. The individual engineering changes (as listed in the Definition of Change) are linked to test case selection by a theoretical examination of how a particular component change may compromise overall performance in a particular

'environment'. The change categories (mapped across each component of the system), will be linked to the individual environments (or test cases) with an 'Applicability Matrix'. Certain changes are likely to compromise the overall performance in certain tests in different ways. For example, a change to the Liquid Crystal Display, illustrated in Figure 8, which has an impact on the 'Thermal Management' change category, is likely to compromise the performance of the system in HOT TEMP TESTING. On the other hand, it will not compromise system performance in DUST TESTING. This knowledge is captured in the applicability matrix and defines how each individual engineering change (as defined in the DoC document) is converted to an individual 'risk' for each usage environment (or test case). The objective was to convert the tacit knowledge into a global theoretical knowledge shared within the company.

5.3 Historical Test Data

One important feature that needs to be captured in the architecture is the ability to access and use the previous project test results data. The analysis of the AS-IS model (as illustrated in

Figure 3) helps to identify that this data was stored locally by each PA in different format, mainly in Excel documents. The key challenge of this portion of data is to categorise the data in such a way that it aligns (and thus can communicate) with the DoC and Theoretical Knowledge data. Therefore, the historical test data (which is also called performance test data) must be structured in such a way that it can be used as a weighting matrix and applied to the engineering changes (as defined by the DoC document) resulting in individual 'risks' for each usage environment (or test case). The new architecture should be able to store all these data in a common database. It has been called “Performance Database” and its standardisation and implementation will be explained in the section 6.2.

5.4 Capturing the data interactions within the Management of Change tool

The Management of Change (MoC) has been created as the core component to capture the data interactions among the different element of the KBE-ProVal Architecture (shown in Figure 4). Within the MoC tool there exists 5 main sources of data, as shown in Figure 5. These are:

1. DoC - Product Specific (PS) Input to the system
2. Historical Test Data (HTD) - Generic (G) weighting matrix
3. Historical Test Data (HTD) - Product Specific (PS) Output of the system
4. Theoretical Knowledge (TK) - Generic (G) weighting matrix
5. Theoretical Knowledge (TK) - Product Specific (PS) Output of the system

The generic data (G) resides in the model permanently. The product specific data (PS) is transient (changes each time the KBE-ProVal is used). The generic data is actually dynamic - the HTD (3) continually evolves each time new test results are entered into the database and the TK (5) is updated 'as-required' as the logic (weighting) is continually optimised. DoC (1) is the only input in the KBE-ProVal. HTD (2) is an output and is the result of a mathematical operation between DoC (1) and HTD (3) - This can be called 'Risk Assessment A'. TK (4) is an output and is the result of a mathematical operation between DoC (1) and TK (5) - This can be called 'Risk Assessment B'.

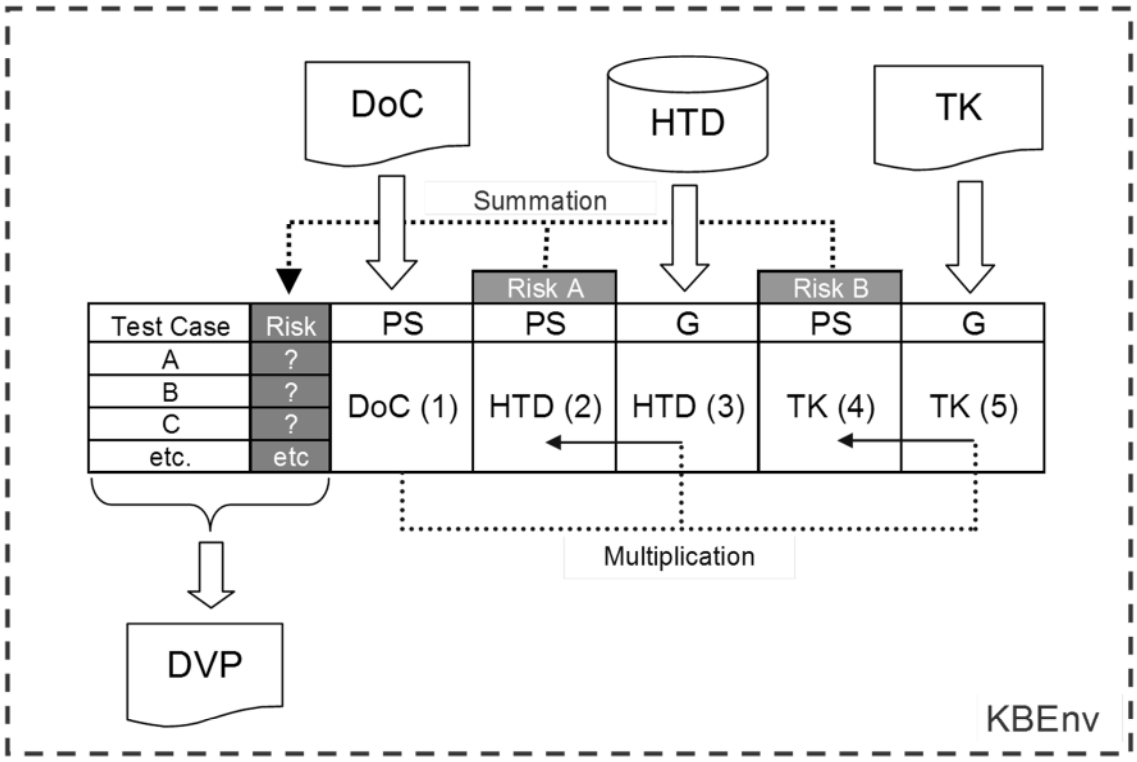


Figure 5: The logical interactions inside the MoC tool

The overall goal of the MoC tool is to assign a risk rating to each individual usage environment or test case, as a result of an engineering change on a specific product. The overall risk for each environment can be considered as the mathematical sum of the two contributing risk assessments (A and B).

According to the logic inside the MoC tool based on the theoretical knowledge, the definition of change information will be processed in order to get an evaluation of the risks associated to each test category. These risks will be summarised in a table to support the PA decision making process. Once the PA makes the decision, a DVP (see

Figure 7) will be automatically created with the test list to be performed.

The following explain the interactions of each key component with the MoC tool as illustrated in Figure 4.

- **Interaction between Theoretical Knowledge – MoC**: the Theoretical Knowledge is the logic inside the MoC tool. Every change in a component triggers an evaluation of the risk for certain tests.

- **Interaction between Historical Test Data – MoC: t**he Historical Data is composed by the previous projects failures data. For a refresh project, most of the components have been already tested. For this reason, previous failures are very good indicators to determine the risk of a component to fail again the same test. The failures need to be analysed and classified in a matrix "components/tests" which gathers all the components that have failed a specific test category in previous projects. This information will be used in the MoC tool to evaluate the risk of the tests selected.

- **Interaction between Definition of change – MoC:** the highlighted changes inside the MoC tool will trigger an evaluation of the risk according only to the design-changed components. This will lead to a restricted list of tests to be performed.

- **Interaction between DVP –MoC**: After processing the MoC tool, a selection of the tests cases to be performed is presented to confirm or change the decision proposed. Then, the DVP will be filled out by the list of tests selected. The list of tests and the sequence to follow may differ according to the customers. Thus, the architecture will help to support and assist the creation of the DVP in agreement of this fact.

- **Interaction between test results – Historical Data:** The test results arriving from the test facilities will be captured and represented in a new standardised way that will allow to be easily inserted in the Historical Data database. This loop in the architecture will lead to a decision making process more and more accurate.

All these interactions have been captured in the KBE-ProVal architecture and its implementation is explained in the following section.

6 KBE-ProVal framework implementation on Teamcenter platform

Teamcenter platform is the Product Lifecycle Management (PLM) software used in the automobile parts supplier company where this work has been developed. For this reason, it has been necessary to analyse and understand the Teamcenter capabilities in order to implement in this platform the KBE-ProVal architecture previously designed.

A PLM tool is a strategy and enabling technique to help a company to succeed in the manufacturing industry. It helps staggeringly to maintain timeliness, validity, accuracy and traceability of product data. Moreover it allows controlling engineering changes caused by different reasons (Zhang et al., 2008).

In the case of the automotive development there are some common product lifecycle stages that can be defined as follows: 1) Requirements development, 2) Engineering design, 3) Manufacturing planning and validation 4) Production and testing, 5) Maintenance and repair.

Siemens Teamcenter Software is a lifecycle management solution (Cable, 2009). It assures that products and projects information flows fluently. It also provides the advantage of allowing the users to interact with Teamcenter from tools and processes they use every day, such as Microsoft Office or Microsoft Outlook. It integrates several options such as file management of products, configuration management, change management, etc and it allows information to be shared across different modules avoiding redundancy (Zhang et al., 2008; Mecham and Hughes, 2008; Maurer, 2009).

Teamcenter provides compatibility, configuration flexibility and good speed request response, satisfying customers' requirements and optimising this way the product life cycle (Shouming et al., 2009). Teamcenter uses CORBA and ODBC technologies to coordinate the enterprise's work platform and it uses C++ programming language to connect to the office packages, CAD software, SAP and other ERP systems. To develop its client it uses Java. TcSE's structure is formed by a user interface layer, an application system layer and a system support layer (Shouming et al., 2009).

Before the implementation in the Teamcenter platform, standardisation of data is needed; that is presented in the following section.

6.1 Definition of Change data standardisation

As Simsion (2005) stated: "People who have been consulted and (better still) who have contributed to the design of a system are more likely to be committed to its successful implementation". The standardisation process has been developed according to this strategy, to get the commitment of the people involved in the process. The standardisation of each element of the KBE-ProVal (as shown in Figure 4) will be detailed in the following sections.

The definition of change (DoC) is one of the key elements as it is the element that will trigger the rest of the process. Its main functions are to display, share and store all the design changes of a product. Mainly a non formal communication among engineers make difficult to trace the data exchanged during the definition of change process.

The DoC is completed by the design engineer's team leader (PDTL), as described in section 4.1. Thus, the PA team needs to understand this DoC. The DoC element has to capture the design changes in the different components of the product and the components attributes affected. This is done by the lead

design engineer. Furthermore an evaluation of the risk based on the experience of the design engineer team has to be specified according to these attributes.

The DoC has got two parts: component categories and the product design change categories. The following employees were interviewed in order to standardise the component categories of the DoC: PA engineers, product design team leaders, and electrical, software and mechanical engineers. A list of components for a specific product in the company was developed according to the following criteria: 1) Accuracy to the real product, 2) Understanding of the end users and 3) Coherence with the framework. Then, a categorisation of the attributes that can be affected by the change of a component was created according to the following criteria:

- Coherence with the test categories (i.e. the test types, such as vibration, hot temperature)
- Coherence with the PA knowledge
- Understanding of the end users (i.e. PA engineers who will use the system)
- Integration in the framework presented in Figure 4.

Finally, a scale to assess the risk associated to each changed component and its attributes was agreed between the PA team and the design engineer's team. The scale ranges from 1, which indicates a low risk design change, to 3, which indicates a very high risk design change (e.g. a major or complete re-design of a component).

6.2 Performance Database implementation

The Performance Database element of the KBE-ProVal is one of the largest parts of the system, as it captures the historical test data (see section 5.3). Its design and implementation has huge effects on the capacity and implementation of the whole system. As it needs to be globally shared, its size needs to be

controlled. Also, its operation will be successful depending on the accuracy of its design. To design the Performance Database, the following actions were performed:

- Identifying the needs of the end user, in this case the PA engineer.
- Understanding this element and its functionality as part of the whole KBE-ProVal. It has to comply with the requirements for integration and guarantee an accurate performance when implemented.

Therefore, the Performance Database was designed to satisfy the following needs:

1. Store in one place all the data from previous projects especially test results.
2. Filter the data according to the user's needs.
3. Optimise the speed when accessing the data in order to save time and increase the process efficiency.
4. Facilitating the communications with the MoC tool

Having considered these needs, the following actions were performed:

1. Standardising reports of test results from previous projects.
2. Capturing general information about previous projects, such as personnel involved, test facilities, etc.
3. Capturing information about product components failures of previous projects.
4. Capturing information about test performance and test results from previous projects.
5. Adapting the data to a format that will allow to be filtered by the user.

The next step consisted of generating and validating a list of possible fields to structure the performance database. As this database has to store the information from the previous projects, the structure has to allow a logical method to be stored and retrieved, related and linked. This database aims

to be used globally. For this reason the standardisation was considered regarding to a future expansion. The Performance Database has been divided in the following four sections: 1) General Information, 2) Component, 3) Test and 4) Failure.

Every field for each section was analysed in detail during several interviews with a PA engineer. A total of 45 fields was the result once all the fields were validated. Regarding to the implementation, for each field was defined the format (text, close lists, dates or numbers). However only two fields are required from this PD, these are: the component categories and the test case. The reason for that is the interest to identify the test failure of the particular component of particular test case from the previous projects.

6.3 Implementation of the MoC tool

Due to the Teamcenter Platform programming restrictions the core of the tool had to be developed in MS Excel and not directly in the main platform. Although it seems an inconvenience, many advantages are obtained:

- The re-programming of the tool can be done easily by a non- TcSE expert.
- The high compatibility of MS Excel with third party applications allows the engineer export the results to other platforms such us Matlab or CAD applications.
- The prebuilt communication protocol between TcSE and MS Excel allows the tool to interact directly with the data inside TcSE.

To develop the MoC tool, three different programming languages were needed:

1. TCL. (Tool Command Language) Four macros have been developed in TCL language. These macros reside inside TcSE platform and enhance the communication between the tool and data stored in TcSE. (TCL Develope XChange, 2010)

2. Excel formulas. The core of the calculations is processed using excel formulas. Array formulas have been crucial in these calculations.
3. Visual Basic Excel macros. This macros has been developed in order to make up for the lack of interactivity of the excel formulas between different sources of data. (Green J. et al. 2007)

The above programming functions result in a powerful tool with a user friendly interface. Figure 6 shows a user interface screenshot of the KBE-ProVal implementation on the Teamcenter Platform . The MoC tool has been developed in excel but keeping the link with all the data in the system. To achieve this, the tool is launched through TcSE so the information previously stored in TcSE is automatically exported to an Excel sheet by a TcSE template developed for this purpose.

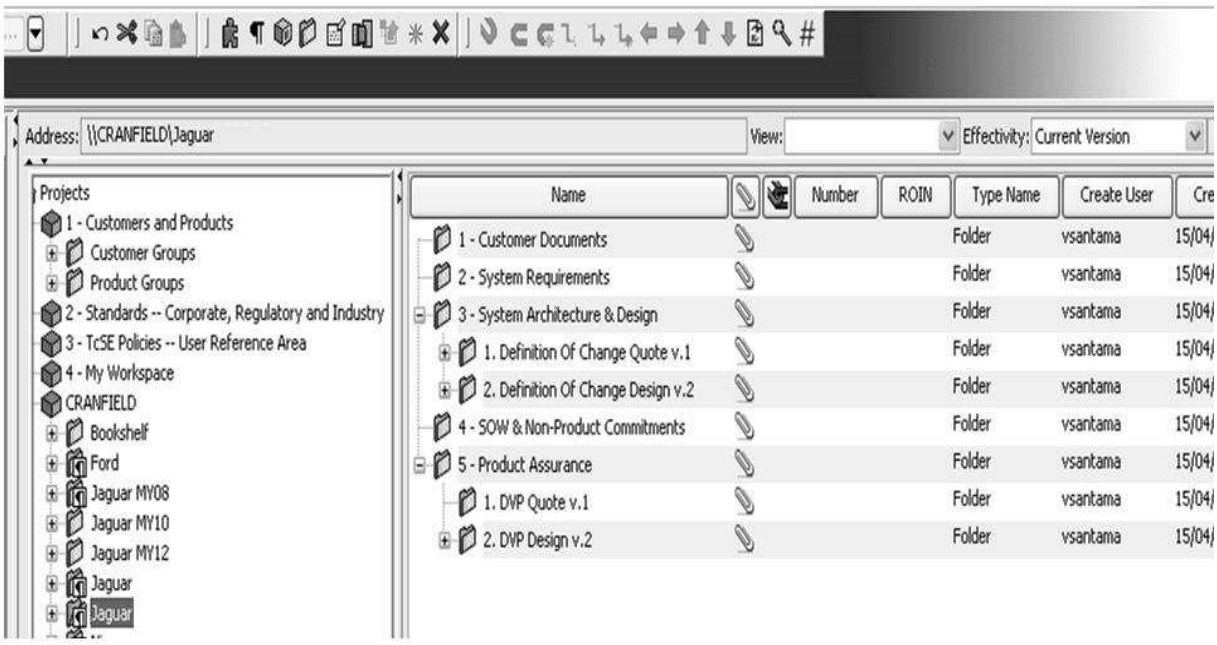


Figure 6: User interface screenshot of the KBE-ProVal implementation on the Teamcenter Platform

To calculate the output, the MoC tool needs to retrieve information from different places: DoC and Historical Test Data. The DoC information is linked with the tool by using another template. The Historical Test Data (or the performance database) is stored in TcSE by using TcSE properties to define the database. Each entry of the database is a standard object of the system called requirement. This

historical data is downloaded and ready to use in a temporal file in the user's computer every time the tool is launched. This is achieved by an activator that is run every time a project is created. This way the tool has an up to date version of the database every time is it used.

Once these two inputs are ready, a first draft of the DVP is proposed. Furthermore, two extra functions have been implemented to customise the results:

- One is a modifiable risk threshold; a value of risk under this threshold will highlight a test with no need to be performed. This function affects the entire test.
- The second is a user adjustable risk that allows the PA engineer to modify each test risk according to the current situation. This function affects only the test adjusted.

The final results for the DVP are shown in real time while modifying the parameters. Finally, when the results are ready, they will be stored in TcSE. This is done by opening another template in excel live. The Excel live functionality allows the simultaneous copy of the data from Excel to TcSE. Several Visual Basic macros have been programmed to enhance the user friendly interface by using dedicated buttons.

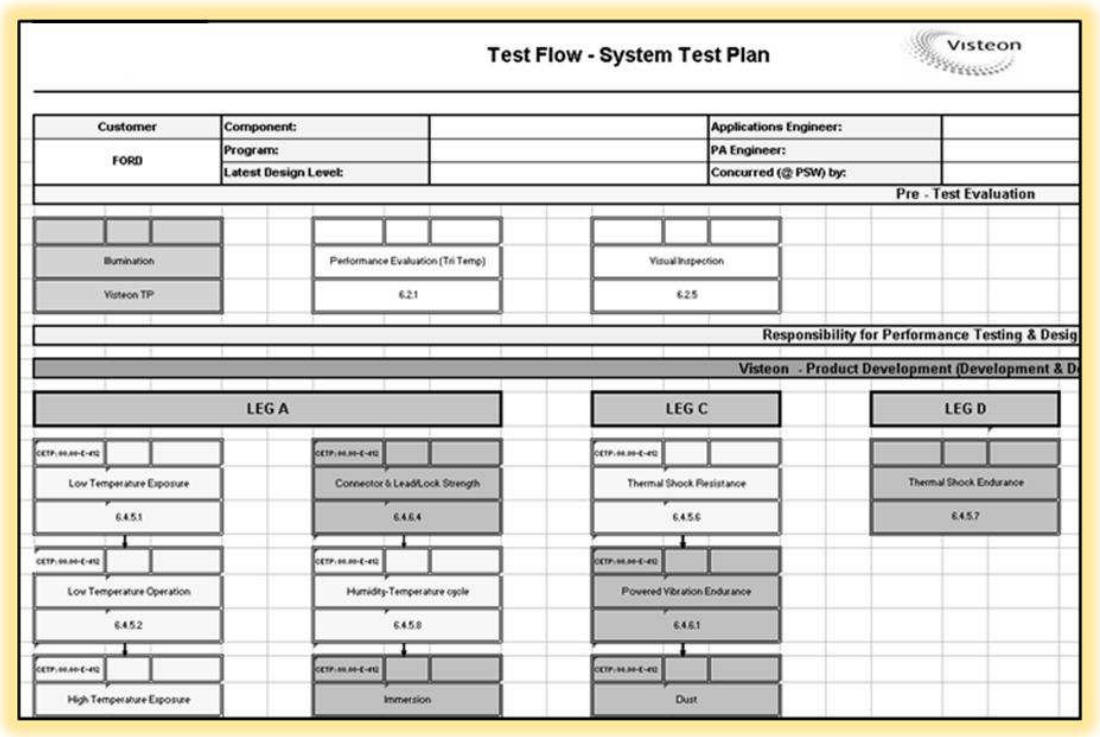


Figure 7: Design validation plan example

When the KBE-ProVal is run completely on the TcSE platform, a first draft of the design validation plan is ready and an example is illustrated in

Figure 7. This action is mainly realised by using excel formulas according to the result of the tool. Conditional formatting function in excel is used to format the test plan in the same way it is done now in the company. After agreement with the customers, the test plan will be sent to the testing facilities.

All the information generated by the tool is stored in TcSE allowing the user to resume the process of design validation of refresh projects using the KBE-ProVal later.

7 Case study validation of a refresh project

Figure 8 shows the Liquid Crystal Display (LCD) in situ on the printed circuit board of the example cluster.

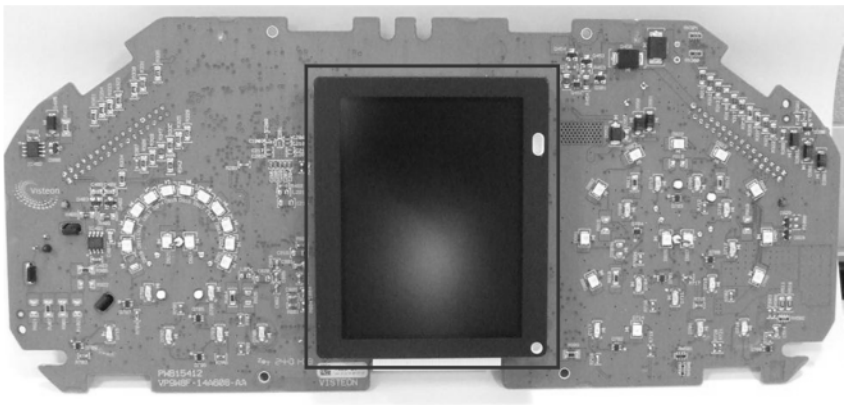


Figure 8: LCD: display of driver information (e.g. radio station)

The mathematical relationships within the MoC tool have been explained in section 5 and this section uses an example to provide some context for the information interactions.

Step 1. **Define Engineering Change**

The display is similar to one already used in the existing product except it runs 5 degrees hotter and is 25grams heavier. This results in an entry in two of the seven available change categories (against just one component in this case, as shown in Figure 9). Mass or Stiffness change is classified as severity ‘3’ (i.e. high risk design change), Thermal Management change is classified as severity ‘1’ (i.e. low risk design change).

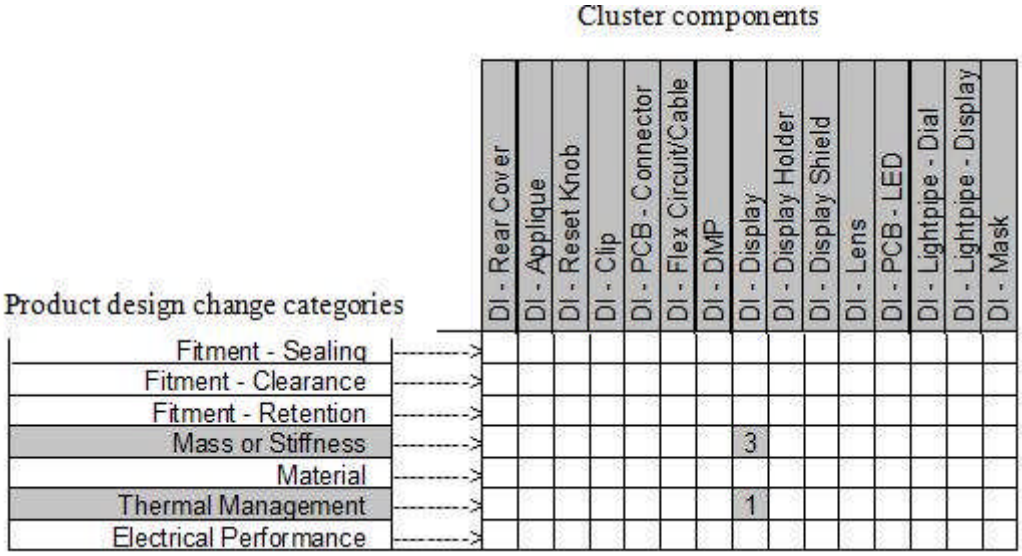


Figure 9: DoC of the example cluster

Step 2 Calculate Risk due to Theoretical Applicability

The proposed engineering changes will result in a different risk rating for each of the test cases, depending the individual applicabilities as defined by the weighting matrix. The weighting matrix for both Theoretical Applicability and Historical Test Data is not shown for simplicity reasons, only the output as a result of mapping with the engineering change (DoC) inputs. This matrix captures the number of failures of a specific test.

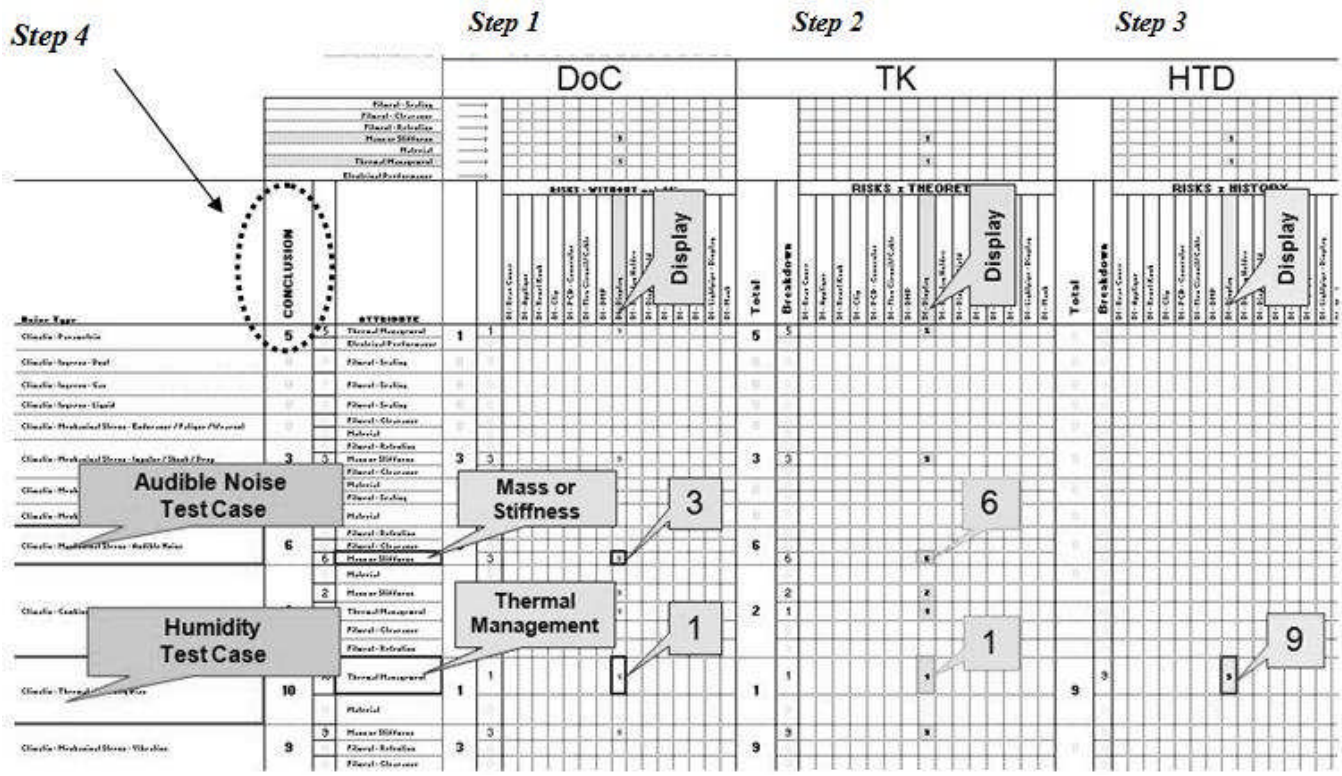


Figure 10: MoC tool of the example cluster

If the Humidity test case is considered in this example (see Figure 10) – a risk of '1' will be identified for this particular test case once the engineering change is mapped to the Theoretical Knowledge matrix. The weighting matrix is key to accurately defining the theoretical risk – this will initially be the responsibility of an experienced engineer to define but as the KBE-ProVal is used more, the matrix will naturally evolve. If the output of the MoC tool (see Figure 10 - highlighted with a dotted oval) does not match expectations - the matrix can be adjusted and the model evolves. In Figure 10, in the step 2-TK '6' is generated as a result of multiplying 3 (high risk design change) by 2 (from the weighting matrix).

Step 3 Calculate Risk due to Historical Test Data

This is similar to the process outlined in step 2 whereas the engineering change is mapped to a weighting matrix except in this case the matrix will be formulated based on real test data. The data is constructed such that the format is common across all matrices. For this example, further risk would only be identified if a failure had occurred on a similar component in a similar test case during previous testing.

In the example shown (see Figure 10) a risk of ‘9’ has been identified for audible noise testing once the engineering change is mapped to the Historical Test Data weighting matrix. This means that this specific failure has been recorded 9 times in the past. If this knowledge (cross platform test data) is not centralised and part of a decision making tool then it is impossible to make decisions on this basis.

Step 4 Review output and Amend

The MoC tool will not always produce a satisfactory output, hence it has been made flexible so that users can override as required. This process is also performed via a multiplication matrix, so that the learning is preserved and made available for administrators to adjust the model as appropriate.

Taking *just two test cases*, we could expect to see a risk applied to both humidity testing and audible noise testing as a result of the one engineering change (see Figure 9) outlined above. In the example the risk is ’10’ for humidity testing and ‘6’ for audible noise testing

8 Conclusions

The enterprises in the present era are leaving no stone unturned to improve their business and customer base. They are going global and also offering customised products to their customers. Looking at their pace, it appears that an important competitive advantage in the future for an enterprise will be the “Knowledge”, that an enterprise produces. This knowledge would be a competitive advantage only when an enterprise can identify, capture and learn from it.

The KBE-ProVal could not have been developed without a detailed analysis of the AS-IS model and without getting the right data and understanding the interaction between them. The software

development has required a team of software developers and a simultaneous validation of the way it was being developed. The conceptual KBEnv has been a very good guide for the later implementation.

Previous projects are the best source of knowledge repositories and this knowledge can be captured for better decision making by supplying logic and facts. Since the project has been developed on the company's existing infrastructure, it has been very cost effective for the company. Also, the framework is highly scalable and can be further extended to be globally used in the company.

Parameter	Before KBE-ProVal	After KBE-ProVal
Communication Time	36 hours	5 hours
Confidence in decision making	72%	90%
Relative standard deviation of DVP	55%	10%

Table 2: Main enhancement indicators

The impact of the KBE-ProVal has been measured against three parameters namely; time spent in communicating with other engineers, confidence in decision making and standardisation of the output (Design Validation Plan- DVP) which are the indicators used within the sponsoring company. Table 2 shows enhancement indicators before and after KBE-ProVal. These results have been obtained via completion of a questionnaire and a case study example with the main stakeholders. For After KBE-ProVal parameters, the developers of the KBEnv estimated the values and these were verified with the expectations of two PA engineers. The communication time between mechanical designers and product assurance has been reduced from 36 hours to an estimated 5 hours. Also, the confidence in decision making by the product assurance engineers regarding the design validation of the refresh project has been estimated to be increased from 72% to 90%. Lastly, the relative standard deviation of DVP has been decreased from 55% to 10%; this is an important factor showing that the differences in the number of tests that the product assurance engineers decide to undertake have been minimised. The relative standard deviation of DVP is likely to be reduced significantly (saving cost and time) as engineers will

become more comfortable with risk taking if they use real historical test data and a globally shared database of theoretical knowledge to justify their decisions when designing a DVP.

The future work entails capturing the historical data of the last two years, which will increase the confidence and the accuracy in calculating project specific risk. In addition, the present work has been tested and used by one product assurance engineer and training is already planned for the rest of the product assurance team. Furthermore, the KBE-ProVal (currently capturing the mechanical design) will include the re-design validation of refresh projects from the point of view of other functional elements of the product, namely electrical, software and illumination.

Acknowledgements

The authors would especially like to specially thank Visteon Engineering Services Ltd. for participating in this piece of research. The research presented in this paper has been conducted as part of a European project entitled ‘Lean Product and Process Development (LeanPPD)’. The project involves multiple research and industrial partners from the UK, Spain, Germany, Italy and Poland. The project is supported by the Commission of European Community, (contract number NMP-2008-214090) under the NMP Programme (Nanosciences, Nanotechnologies, Materials and new Production Technologies). The authors wish to acknowledge the European Commission for its support as well as the other partners in the consortium (http://www.leanppd.eu).

References

Bukowitz, W. a. (2000), "The knowledge management fieldbook". London: Prentice Hall.

Cable, J. (2009), Teamcenter 8 Aims to Drive PLM Productivity, *Industry Week/IW*, vol. 258, no. 8, pp. 42-44.

Chen, Y., Chen, Y., and Wu, M. -. (2009), "Development of a distributed product knowledge service system", International Conference on Complex, Intelligent and Software Intensive Systems, CISIS 2009, 16 March 2009 through 19 March 2009, Fukuoka, pp. 136.

Cowley-Durst, B. (1999) "Gathering Knowledge for Your Knowledge Management Sytem", International Society for Performance Improvement, pp. 23-27.

Dalkir, K. (2005), "Knowledge Management in Theory and Practice", USA: Elsevier Publications.

Elias M. Awad, H. M. (2004), "Knowledge Management. Pearson Education Inc.", Prentice Hall.

Green J., Bullen S., Bovey R. and Alexander M. (2007), “Excel®2007 VBA Programmer’s Reference” Wiley Publishing, Inc.

H.J. Müller, A. S. The Knowledge Factory - A Generic Knowledge Management Architecture. Germany.

Hamid R. Parsaei, S. K. Manufacturing Decision Support Systems. Chapman & Hall.

Jennex, M. E. (2005). "Case Studies in Knowledge Management". London, Hershey: Idea Group Publishing.

Jinette de Gooijer (2000), "Designing a Knowledge Management Performance Framework", Journal of Knowledge Management, vol. 4, no. 4, pp. 303-310.

K. Rodriguez, A. Al-Ashaab, (2004), "A review of internet based collaborative product development systems. in: Proceedings of the International Conference on Concurrent Engineering: Research and Applications" Cranfield University, UK

Maurer, T. (2009), "Teamcenter's data accuracy shortens HP printer design cycle. (cover story)", Electronic Engineering Times (01921541), , no. 1566, pp. 26-28.

McElroy, M. (1999), "The Knowledge Lifecycle", ICM Conference, Florida, Miami, US.

McInerney, C. (2002), "Hot topics: knowledge management – a practice still defining itself", Bulletin of the American Society for Information Science, vol. 28, no. 3, pp. 14-15.

Mecham, M. and Hughes, D. (2008), "Boeing Taps Siemens PLM", Aviation Week & Space Technology, vol. 169, no. 6, pp. 64-65.

Meyer, M. Z. (1996). "The design and implementation of information products", Sloan Management Review , pp. 43-59.

Michael A. Carrico, J. E. Building Knowledge Systems . New York: Intertext Publications, McGraw-Hill Book Company.

Morey, D. (1998), "Knowledge Management Architecture", Retrieved April 21, 2010, from www.brint.com: http://www.brint.com/members/online/120205/kmarch/kmarch.html

Nonaka, I. (1994), "A Dynamic Theory of Organizational Knowledge Creation", Organizational Science, vol. 5, no. 1, pp. 14-37.

Ranjit Bose (2004), "Knowledge Management Metrics", Industrial Management & Data Systems, vol. 104, no. 6, pp. 457-468.

Schmidt, D.C., Gokhale, A., Natarajan, B. (2004), “Leveraging Application Frameworks”, pp. 73.

Shen, H., Wall, B., Zaremba, M., Chen, Y. and Browne, J. (2004), "Integration of business modelling methods for enterprise information system analysis and user requirements gathering", Computers in Industry, vol. 54, no. 3, pp. 307-323.

Shouming, H., Yongxian, L., Enchao, Y. and Zhongqi, S. (2009), "Research on rapid response design system based on Teamcenter Engineering software", Vol. 4, pp. 255.

Simsion, Graeme C. (2005), "Data modeling essentials", Elsevier.

TCL Develope XChange, http://www.tcl.tk/ (accessed 22nd March 2010).

Tom Davenport, L. P. (1998), "Know What You Know", CIO Magazine .

Wiig, K. (1993), "Knowledge Management foundation", *Schema Press*, Arlington, Texas, US.

Zhang, Y., Bai, X., Xie, H. and Liu, Y. (2008), "Research on modular innovation system for lifecycle", pp. 1688.

Zheng, L. Y., McMahon, C. A., Li, L., Ding, L. and Jamshidi, J. (2008), "Key characteristics management in product lifecycle management: a survey of methodologies and practices", Proceedings of the Institution of Mechanical Engineers -- Part B -- Engineering Manufacture, vol. 222, no. 8, pp. 989-1008.



https://dspace.org/

Knowledge based environment to support product design validation

Al-Ashaab, Ahmed; Molyneaux, M.; Doultsinou, Athanasia; Brunner, B.; Martínez, E. ...

2012-02-01
Elsevier
https://doi.org/10.1016/j.knosys.2011.06.019

Downloaded from a DSpace repository. https://dspace.org/