COMPLEX PRODUCT FAMILIES MAINTAINABILITY ASSESSMENT AT DESIGN STAGE

SERIGNE DIAGNE
University Assane SECK of Ziguinchor / LI3, Ziguinchor, Senegal
sdiagne@univ-zig.sn / serigne.diagne@insa-strasbourg.fr (Corresponding author)

AMADOU COULIBALY
INSA of Strasbourg / LGeCo, Bas-Rhin, France
amadou.coulibaly@insa-strasbourg.fr

FRANÇOIS DE BERTRAND DE BEUVRON
University of Strasbourg / I3, Bas-Rhin, France
francois.debertranddebeuvron@insa-strasbourg.fr

ABSTRACT

The functionalities and performance expected to modern products are increasing continuously. They generate very important conceptual, technological and functional complexities and require designers to study the behaviour of the products at the early design stage. Moreover, in order to model complex product families and extract from their instances the ones that fulfil the requirements, constraints and specifications, it is necessary to be able to instantiate the meta-model and evaluate the resulting instance (product). This instance is then either validated or improved or abandoned according to the satisfaction of the requirements. It is also possible to switch to another instance and to restart the process. In this paper, we propose an improvement of the maintainability indicators such as the disassembly time and the replacement time of failed components and a behavioural performance engineering algorithm for complex product families at the early design stage. This methodology allows us to extract from complex product families, the ones that satisfy the requirements and constraints of the customers and designers.

Key words: Behavioural performance engineering, Complex product family, Maintainability, Product-BPAS.

INTRODUCTION

The design of complex product families is becoming increasingly important in research in recent years. It generates a significant number of products that can meet requirements. The products must be evaluated at the early design stage in order to make a decision (approval, improvement or abandonment) to reduce the delay ant the cost. Thus, designers must be able to identify in the product families the products that meet the requirements of the customers. In our previous work, we have proposed a conceptual semantic design approach and a conceptual semantic modelling methodology allowing us to model complex product families (Diagne *et al.*, 2014a)(Diagne *et al.*, 2014b). Furthermore, many recent investigations focussing on the evaluation of behavioural performance of complex products at the early design stage are conducted by our research team. Then, metrics that can be assessed at the early design stage and formulas to calculate them are proposed: maintainability and safety indicators (Coulibaly *et al.*, 2008a), reliability indicator (Zwingmann *et al.*, 2002), safety indicator based on availability (Houssin *et al.*, 2014). Similarly, a complex mechanical products maintenance document (Coulibaly *et al.*, 2008b), an optimal

disassembly sequence strategy (Zwingmann *et al.*, 2008), a mathematical model for the minimization of the average time of disassembly (Menye *et al.*, 2009), a recycling-oriented design (Coulibaly *et al.*, 2010), etc. have been proposed. Other research teams have proposed models and indicators to improve the behavioural performance of complex products at the design stage. Thus, an approach for the prediction of the reliability of new products has been proposed by Liu *et al.* (2013). In order to create a method allowing the reduction of the cost of the maintenance of complex systems, Poncelin *et al.* (2008) have proposed a reliability-oriented design methodology. An approach for the maintainability of complex products based on fuzzy logic have been proposed by Slavila *et al.* (2005). Ding (2009) has proposed a design method based maintainability by integrating the model of characteristics and the maintainability design criteria with measurement indices. To improve the maintainability and the recyclability of complex products, the design for assembly (DFA) and the design for disassembly (DFD) have been the subject of many research (Boothroyd *et al.*, 1992)(He *et al.*, 1997)(Desai *et al.*, 2003)(Güngör, 2006)(Bogue *et al.*, 2007). In this sense, the modularity-based design and the assembly-based design are coupled by Abdullah *et al.* (2006). In their work, a comparison between these two kinds of design have been done before coupling them to improve the maintainability of complex products. In all these work, authors have proposed indicators, formulas and/or models for the evaluation of the behavioural performance of complex products at the early design stage. However, a performance engineering methodology allowing to identify within a set of products (families of products) the ones that fulfil the requirements for the behavioural domains: reliability, availability, safety, maintainability, recyclability and the constraints at the interface of these domains (multi-domain evaluation) still to be defined. The first contribution of this paper is an improvement of the assessment of the disassembly time and the replacement time of failed components in the instances of complex product families. The second contribution is an algorithm for behavioural performance engineering of complex product families.

To achieve these goals, we present the formulas to estimate the disassembly time and the replacement time of failed components in Section 2. The algorithm giving the process of the behavioral performance engineering of complex product families is explained in Section 3. Finally, the conclusion and the direction of our future work are given in Section 4.

COMPLEX PRODUCT FAMILIES MAINTAINABILITY EVALUATION

Disassembly indicator

Accessibility and position of the components

The disassembly time corresponding to a sequence is often assessed as the sum of the time (or difficulty) required to remove each link. A disassembly sequence is a series of components and associated links in the right order to disassemble a product. Table 1 gives the values of the parameter *q* which indicates the level of difficulty to remove each kind of link (its weight). More a link (L_i) is difficult to remove, more its weight (q_i) is high.

Table 1: Link's weight according to the difficulty to remove them (Coulibaly et al., 2008a)

Link topology	Link weight
No contact	$q_1 = 0$
Contact	$q_2 = 1$
Clipping	$q_3 = 2$
Screwing	$q_4 = 3$
Bolting	$q_5 = 4$
Riveting	$q_6 = 6$
Housing	$q_7 = 7$
Gluing	$q_8 = 8$
Welding	$q_9 = 10$

Using the weight of the links (q_i / *1 <= i <= 9)*, the time required to disassemble a product (D_d) is obtained from the links (L_i / 1 <= i <= n) that assemble its components. Equation 1 gives the formula to assess
D_d:

$$D_d = \sum_{i=1}^{n} m_i q_i \qquad (1)$$

m_i is the number of occurrences of the link L_i and n is the number of links within the sequence.

In addition to the parameter *q*, we add two other parameters:

i. the parameter *p* gives the position of the next link in the sequence;
ii. the parameter *a* gives the accessibility of this link in the sequence.

These two parameters are described as follows:

i. If the next link in the sequence can be removed from:
   - the current component: p = 0;
   - a neighbour of the current component: p = µ;
   - a n^{th} component from the current component: p = nµ
ii. If the next link in the sequence is:
   - easily accessible: a = 1;
   - not easily accessible: a = α. According to the structure of the product, we can have many levels of difficulty.

The disassembly indicator (D_d) is obtained using the parameters *q*, *p* and *a* as in Equation 2:

$$D_d = \sum_{i=1}^{n} m_i a_i q_i + \sum_{j=2}^{n} p_j \qquad (2)$$

In this formula, for each link L_i its parameter q_i that gives the level of difficulty to remove it is multiplied by its accessibility a_i and its number of occurrences m_i. The time to remove a link is

proportional to its accessibility. After removing a link, if it is not the last in the sequence, the position of the following link is added. It should be noted that the accessibility of a link may change during the disassembly process. An unaccessible link can become accessible after removing other links. Similarly, the position of the next link depends to the current link.

Simultaneous sub-sequences (SSS)

During the disassembly process, it is possible to have a set of successive links in the sequence that can be removed simultaneously. We name such a set simultaneous sub-sequence (SSS). We assume that the necessary tools are available. We denote t_i the time required to remove the link L_i and λ_j the time required to remove the x links in the SSS noted S_j. Then, if we have:

i. x workers, the time required to remove the x links within the SSS is equal to the higher time in the set containing the x removal times as shown in Equation 3:

$$\lambda_j = max(t_1, t_2, \ldots, t_x) \quad (3)$$

ii. y workers (y < x), the time required to remove the x links in the SSS is obtained using Equation 4:

$$\lambda_j = \sum_{i=1}^{\lfloor \frac{x}{y} \rfloor} max(t_{y*(i-1)+1}, \ldots, t_{y*i}) + max(t_{y*\lfloor \frac{x}{y} \rfloor+1}, \ldots, t_x) \quad (4)$$

$$t_i = m_i * a_i * q_i \quad (5)$$

In a sequence S with s simultaneous sub-sequences (S_1 to S_s) with times λ_1, λ_2, …, λ_s and r other links, that do not belong to these sequences, if t_1, t_2, …, t_r, are the time required to remove the r links, the disassembly indicator (D_d) is obtained using the Equation 6:

$$D_d = \sum_{i=1}^{s} \lambda_i + \sum_{j=1}^{r} t_j + \sum_{k=2}^{s+r} p_k \quad (6)$$

If n_i is the number of links within the simultaneous sub-sequences S_i, Equation 7 gives the value of r:

$$r = n - \sum_{i=1}^{s} n_i \quad (7)$$

i. n is the number of links in the sequence S;

ii. n_i is the number of links in the simultaneous sub-sequence S_i;

iii. p_k is the position of the following link L_k, or the position of the farthest link within the following SSS.

For each SSS (S_i) the time λ_i depends on the number of workers and the order of the links. However, if the number of links is less or equal to the number of workers, the value of λ_i is equal to the highest value of t_j. In this case the order of the links is not significant.

In the following Section, we use the times t_j and λ_i to calculate the time required to replace a failed component.

Assessment of the replacement time of failed components

The replacement time (T_i) of a failed component (C_i) is the sum of its dismantling time (T_{di}) and the assembly time of the new component that replaces it (T_{mi}).

If

- E_i with the cardinality h_i is the sequence of components containing C_i and all components that must be removed to achieve C_i;
- LE_i is the set of links containing all links between the h_i components in E_i;
- T_{di} the time required to remove all links in LE_i (the dismantling time).

Equation 8 gives the formula for the assessment of the dimantling time T_{di}:

$$T_{di} = \sum_{j=1}^{s_i} \lambda_j + \sum_{l=1}^{r_i} t_l + \sum_{k=2}^{s_i+r_i} p_k \qquad (8)$$

In Equation 8,

- s_i is the number of simultaneous sub-sequences within LE_i;
- r_i is the number of links within LE_i which are in any SSS.

For simplifying, we consider that the dismantling time (T_{di}) of a failed component is equal to the time for remounting the one that replace it as shown in Equation 9:

$$T_{di} = T_{mi} \qquad (9)$$

Then the replacement time T_i can be assessed using Equation 10:

$$T_i = T_{di} + T_{mi} = 2 * T_{di} \qquad (10)$$

In Equation 9, If we replace the dismantling time (T_{di}) with its formula in Equation 8, we have Equation 11:

$$T_i = 2 * \left(\sum_{j=1}^{s_i} \lambda_j + \sum_{l=1}^{r_i} t_l + \sum_{k=2}^{s_i+r_i} p_k\right) \qquad (11)$$

The behavioural performance engineering algorithm proposed in the following section uses the indicators for maintainability, availability, reliability, safety and recyclability to identify the instances of complex product families that satisfy the requirements.

BEHAVIOURAL PERFORMANCE ENGINEERING ALGORITHM

The behavioural Performance Engineering is an approach for the identification and the optimization of the products that satisfy the requirements and constraints of the customers in a set of products (families of products). It is based on three concepts Evaluation-Optimization-Validation (EOV). It also allows the comparison of two products. The behavioural performance engineering begins with the

choice of the behavioural domains (reliability, safety, availability, maintainability, recyclability). The input of the algorithm are the semantic data extract from the Multi-Solution eXtended Conceptual Design Semantic Matrix MSX-CDSM (Diagne *et al.*, 2014b) that represents several product families. The behavioural performance engineering algorithm includes five main steps (Figure 2):

i. extraction of the data from the matrix that represent the instance selected:
    a. Choice of a family of product;
    b. choice of an instance in this family;
    c. extraction of the data in the matrix representing the instance selected.
ii. choice of the behavioural domain(s);
iii. assessment of the metrics of the chosen domain (s);
iv. validation of each behavioural domain;
v. validation of the instance (multi-domain validation).

For an instance of a family, the steps iii. and iv. can be repeated as many times as we have domains to be evaluated. If the instance does not satisfy the requirements, either it is improved and the process starts by the step i.c., or it is abandoned and saved and another instance is chosen. If the instance satisfies the requirements for each domain or if it satisfies the multi-domain's evaluation, it is retained and saved. It is possible, if several behavioural domains are evaluated for the same instance, to compromise during the multi-domain validation. Indeed, an instance can satisfy the requirements of some behavioural domains but not those of the other. In this case, the designers must make a decision according to the multi-domain's rules. Figure 1 shows the process for the validation of an instance for a behavioral domain.

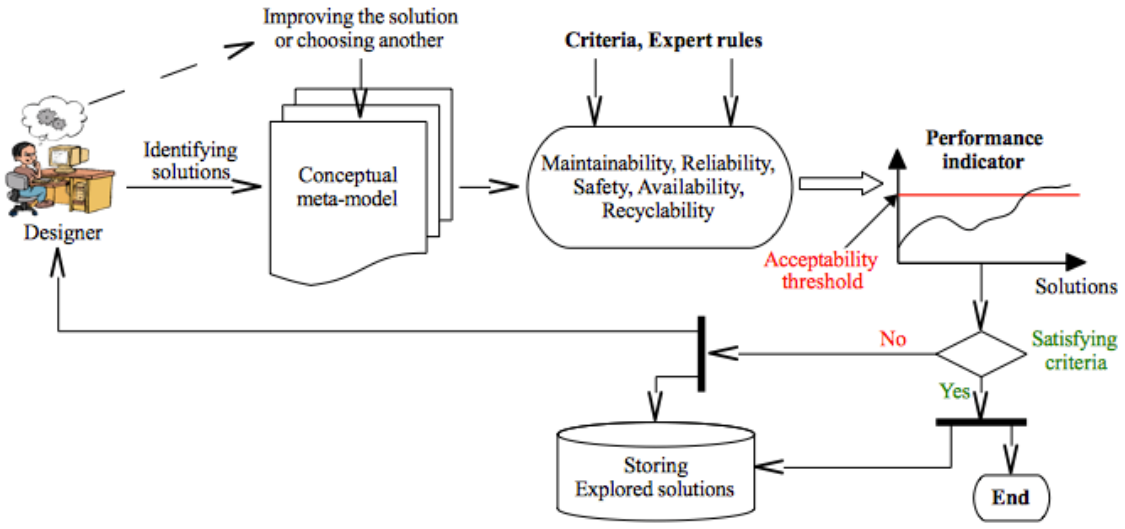

Figure 1: Validation process of an instance for a behavioural domain

Figure 2 gives the algorithm for the behavioural performance engineering process. This algorithm allows to go from a set of classes of solution (families of products) to their instances that satisfy the behavioural requirements. The input of the algorithm are the data extract from the matrix MSX-CDSM that represent the set of classes of solution. A class of solution represented by an eXtended Conceptual Design Semantic Matrix (X-CDSM) (Diagne *et al.*, 2014b) is chosen is this set of classes of

solution. The X-CDSM is instantiated to get an instance (product) of the selected class. The behavioral domains that must be evaluated are selected and the instance is validated for each domain. Indeed, the domains are chosen one by one and for each of them, the necessary data for its evaluation are extracted from the matrix of the Instance. The metric of this domain is calculated and the result obtained is compared to the threshold of this domain (Figure 1). The same process is repeated for the other domain selected. After the validation of the domains one by one, a multi-domain's validation is performed to determine if the instance will be retained, improved or abandoned. This instance is saved with the results of the validations in a database regardless the final decision. If the instance is abandoned, either a new instance of the same class of solution, or an instance of another class of solution is selected. If the instance is retained, it is also possible to choose another instance (in the same class or in another class) for its evaluation. If it is retained, it can be compared to the instances already retained in order to select the best instance(s). The designers can stop the process at any time.

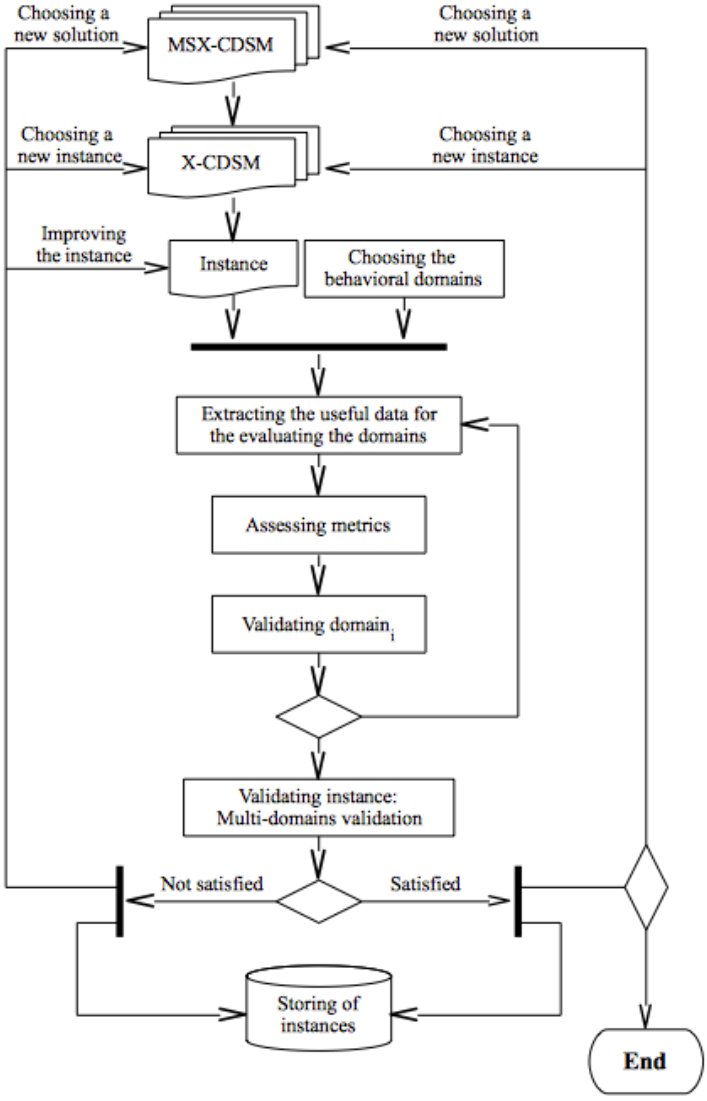

Figure 2: Behavioural performance engineering algorithm

Using this method, the instances of product families are evaluated and validated before the manufacturing phase. This methodology is being implemented in a software for the behavioural performance engineering of complex products. This software is named Product Behavioural Performance Analysis System (Product-BPAS). Its graphical interface is represented in Figure 3. It will be interfaced with classical CAD softwares for the behavioural performance engineering of mecatronics products. The API allowing to add this software in CAD systems as an add-on or a standalone is under development.

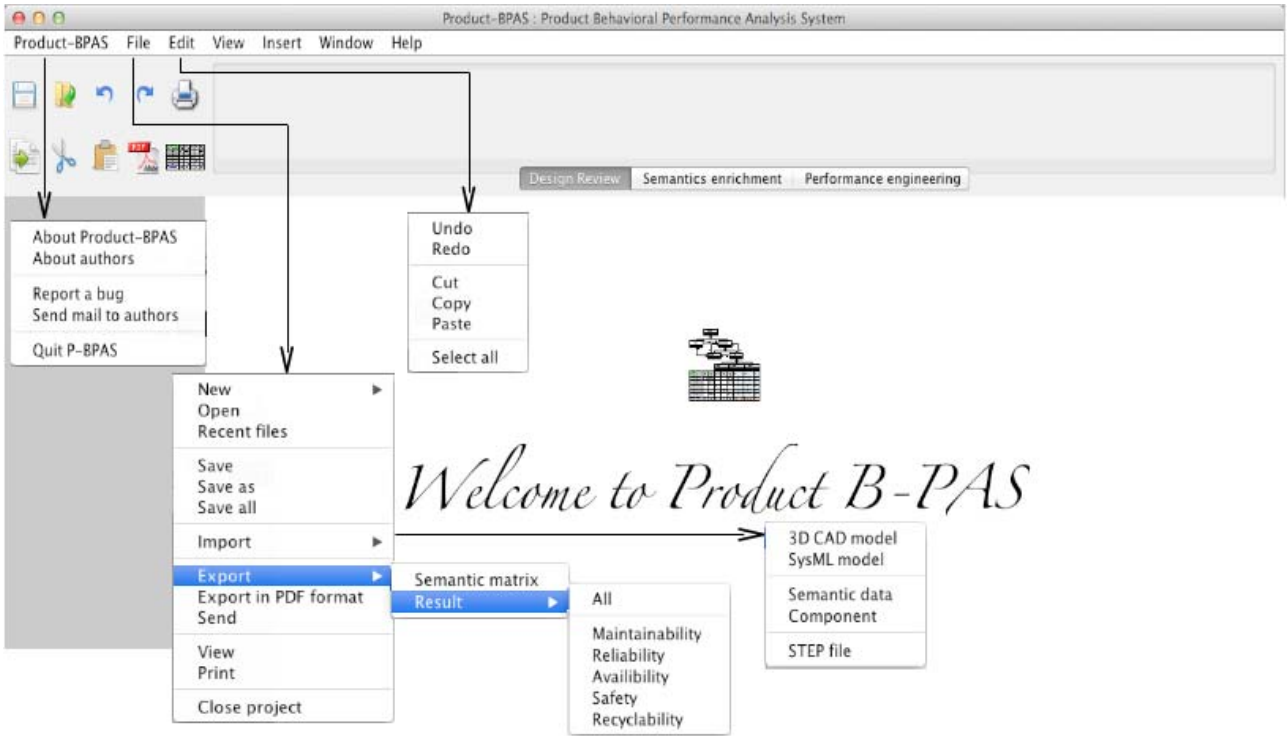

Figure 3: Graphical interface of the Product Behavioural Performance Analysis System

CONCLUSION

In this paper we propose an improvement of maintainability indicators such as disassembly time of complex products families and replacement time of failed components. In the formulas proposed, we have added new parameters such as the position of the links and their accessibility. We also take into account the possibility to remove several links simultaneously. An algorithm for complex products behavioral performance engineering is proposed. It allows to identify in the instances of a MSX-CDSM model representing several families of complex products the ones that satisfy the requirements of the customers. It allows also to compare the instances of the same family of products and the instances of different families of products. The algorithm and formulas proposed in this paper is being implemented in a software named Product-BPAS.

REFERENCES

Abdullah, B., Yusoff, M.S., and Ripin, Z.M., (2006), Integration of design for modularity and design for assembly to enhance product maintainability. In Proc. 1St Int. Conf. 7Th AUN/SEED-Net Fieldwise seminar on Manufacturing and Material Processing, pp. 263-267, University Malaya: Kuala Lumpur, March 14-16.

Bogue, R., (2007), Design for disassembly: a critical twenty-first century disciple. Assembly Automation, 27(4), 285-289.

Boothroyd, G., and Alting, L., (1992), Design for assembly and disassembly. CIRP Annals-Manufacturing Technology, 41(2), 625-636.

Coulibaly, A., Houssin, R., and Mutel, B., (2008), Maintainability and safety indicators at design stage for mechanical products. Computer In Industry Journal, 59(5), 438-449.

Coulibaly, A., Huang, Y., Gardoni, M., and Maranzana, R., (2008), Multi-users dynamic maintenance document for complex mechanical products. Application for automobile. In Proc. ASME, DETC'2008, pp. 125-133. New York City: USA. August 3-6.

Coulibaly, A., Gardoni, M., Gaye, kh., and Huang, Y., (2010), Recyclability oriented lifecycle design for mechatronics products. In Proc. 15Th Design Manufacturing and the Lifecycle Conf., 6, pp. 135-143, Montreal: Quebec: Canada, August 15-18.

Desai, A., and Mital, A., (2003), Evaluation of disassemblability to enable design for disassembly in mass production. Int. Journal of Industrial Ergonomics, 32(4), 265-281

Diagne, S., Coulibaly, A., and De Beuvron De Bertrand, F, (2014), Towards a conceptual semantic design for mechatronic product's family development. In Proc. Int. Conf. On Innovative Design and Manufacturing (ICIDM), pp. 94-99. Montreal: Quebec: Canada. August 13-15.

Diagne, S., Coulibaly, A., Sene, M., and De Beuvron De Bertrand, F, (2014), Complex mechatronic product modeling using a multi-solution, multi-instance extended design semantic matrix. In Proc. 16Th Int Dependence and Structured Matrix Conf. (DSM Conf.), pp. 85-94. Paris: France. July 2-4.

Ding, Y., (2009), Product maintainability design method and support tool based on feature model. Journal of Software Enginnering & Applications, 2(3), 165-172.

Güngör, A., (2006), Evaluation of connection types in design for disassembly (DFD) using analytic network process. Computers & Industrial Engineering Journal, 50(1), 35-54.

He, D.W., and Kusiak, A., (1997), Design of assembly systems for modular products. Robotics and Automation, IEEE Transactions on, 13(5), 646-655

Houssin, R., and Coulibaly, A., (2014), Safety-based availability assessment at design stage. Computers & Industrial Engineering Journal, 70, 107-115.

Liu, Y., Huang, H.Z., and Ling, D., (2013), Reliability prediction for evolutionary product in the conceptual design phase using neural network-based fuzzy synthetic assessment. Int. Jounal of Systems Scienc, 44(3), 545-555.

Ményé, J.B., Ait-Kadi, D., Coulibaly, A., and Caillaud, E., (2009), Mathematical model for the minimization of the mean disassembly time of a mechanical system at design stage. In Proc. Int. Conf. Computers and Industrial Engineering, CIE39, Troyes: France. July 6-8.

Poncelin, G., Derain, J.P., Cauvin, A., and Dufrène, D., (2008), Development of a design-for-reliability method for complex systems. In Proc. IEEE Reliability and Maintainability Symposium, pp. 175-180, Las Vegas: NV, January 28-31.

Slavila, C.A., Decreuse, Ch., and Ferney, M., (2005), Fuzzy approach for maintainability evaluation in the design process. Concurrent Engineering Journal, 13(4), 291-300.

Zwingmann, X., Ait-Kadi, D., Coulibaly, A., and Mutel, B., (2002), Product reliability assessment using virtual samples. In Proc. IEEE Int. Conf. On Systems, Man and Cybernetics, Hammamet: Tunisie. October 6-9.

Zwingmann, X., Ait-Kadi, D., Coulibaly, A., and Mutel, B., (2008), Optimal disassmbly sequencing strategy using constraint programming approach. Journal of Quality in Maintenance Engineering, 14(1), 46-58.