Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 34 (2015) 250 – 255

www.elsevier.com/locate/procedia

9th International Conference on Axiomatic Design – ICAD 2015

Development of a custom software for processing the stress corrosion experimental data through Axiomatic Design

Andrea Girgenti[a]*, Alessandro Giorgetti[a], Paolo Citti[a], Marco Romanelli[b]

[a]*Università Guglielmo Marconi, via Plinio 44, 00193 Rome, Italy*
[b]*GE Oil & Gas Nuovo Pignone S.r.l., via Felice Matteucci 2, 50127 Florence, Italy*
* Corresponding author. Tel.:+39- 06- 377-251; fax:+39-06 -377-25 647.*E-mail address:* a.girgenti@unimarconi.it

Abstract

The economical sustainability and the integrity of oil field equipment depend on the choice of the best material for the working conditions. In oil wells, many environmental corrosion phenomena take place and affect the structural integrity of metallic parts and equipment. In order to investigate material properties in presence of corrosive environments and applied stresses, many laboratories have arisen and many experimental techniques have been developed and setup in order to collect data about the behaviour of corrosion resistant alloys. Available data are traditionally gathered into text files which are difficult to analyze and process since a widespread commercial software does not exist for automated data processing and reporting. Each laboratory has to develop their own tools in order to perform the data post processing for stress corrosion experiments. In this paper, the Axiomatic Design method has been employed in order to develop a custom software for data post processing of the stress corrosion tests. The application of the Axiomatic Design has a key role in defining the software architecture, avoiding useless solutions and improving code optimization from the earliest phases.

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design
*Keywords:**Software design, Axiomatic Design, data post-processing***

1. Introduction

The stress corrosion cracking mechanism is a critical issue in modern oil and gas industry since it affects a huge part of the amount of equipment in oil wells. This is a kind of corrosion which takes place especially into aqueous environments where the presence of some raw chemicals and mechanical stresses can create harmful working conditions. Typical chemicals in oil fields, which are able to make the environment more aggressive, are hydrogen sulphide, carbon dioxide (or a mixture of both) and chlorides. These agents typically make the pH lower, producing the aggression of the outer surfaces of metallic parts, moreover originating secondary effects like the hydrogen embrittlement [1].

The hydrogen embrittlement consists in the permeation of atomic or molecular hydrogen through the metallic surfaces, making the material brittle due to the increasing pressure of the stored up gas within the lattice [2-5]. This effect, in addition to high stresses due to normal operations, may lead to a sudden and expensive failure in well equipment. However, even if the presence of both stresses and aggressive agents is the main cause of the stress corrosion cracking phenomenon, there are many other parameters which can affect the behavior of a given steel or corrosion resistant alloy. Some of these physical parameters are the hardness distribution, the microstructures, the material chemistry, the heat treatments, the site temperature and pressure. This large amount of influential parameters makes difficult to forecast the cracking phenomenon development and its feed rate [6].

The economy and safety of oil fields require an accurate definition of safe working boundaries for several alloys through experimental campaigns using industry standard evaluation techniques.

Common test methods are based on the application of a constant load to the test specimen while it is sunk into a corrosive brine, bubbled with hydrogen sulphide or other gasses. These kinds of test record the time to failure of several stressed specimens within the maximum duration of the test,

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design
doi:10.1016/j.procir.2015.07.064

(that is 720 hours) in order to locate the threshold value for stresses, below which failures do not occur. The stress corrosion tests need a tool to manage and process data. The fulfillment of this need requires a custom software which the authors are going to develop through the Axiomatic Design Theory.

Nomenclature

d	diameter of the specimen gauge section [mm]
$\bar{F}$	measured force array [N]
l_i	gauge length of the i-th element of the measured elongation array [mm]
$\bar{l}$	gauge elongation array [mm]
l_0	initial gauge length [mm]
$\bar{\varepsilon}$	measured strain array
$\bar{\varepsilon}_{el}$	strain values whose the shift from the linear field is within 0,2%
ε_i	i-th element of the measured strain array
ε_y	yield strain
ε_{max}	maximum strain
ε_r	ultimate strain
$\bar{\sigma}$	measured stress array [N/mm^2]
σ_y	yield stress [N/mm^2]
σ_{max}	maximum stress [N/mm^2]
σ_{max}	ultimate stress [N/mm^2]

2. The slow strain rate test technique

As previously stated, the common test methods may require a maximum time of almost one month (720 hours) and need a certain amount of specimens, grouped in samples which are loaded at different percentages of the yield strength in order to find the threshold value for the material safe employment.

Such cycle time may be considered long, especially for those cases where a choice among some alternatives shall be made and insufficient resources are available for comprehensively testing different available specimens, loads and alloys. The slow strain rate technique satisfies this need since it allows to screen a group of available materials in order to identify the one which matches better with the service conditions in a shorter time than the traditional constant load methods. According to [7, 8] one of the main causes of the slow strain rate test success in past years is the relative simplicity and fast execution in order to quickly construct a ranking among the most performing alloys. In slow strain rate tests the object is producing stress corrosion cracks that are metallographically indistinguishable from those obtained through constant load or sustained load methods. The severity of this kind of test is increased in respect with constant load methods due to the effect of the constant strain rate. Commonly used values for the strain rate are from $2{,}5 \cdot 10^{-9} m/s$ to $2{,}5 \cdot 10^{-7} m/s$ therefore, since the crack propagation velocity usually wavers between $10^{-3} mm/s$ and $10^{-6} mm/s$, failures in laboratory test specimens of usual dimensions occur in a much shorter time [9],typically less than 300 hours.

The slow strain rate tester is composed by an autoclave which stores the specimen and the corrosive solution, and a mechanism able to apply a variable deformation with a constant strain rate. The constant strain rate is achieved through a geared servo-motor which is connected to a linear actuator in order to control the dynamic load applied to the specimen.

The loading of the specimen happens since its ends are respectively linked to the fixed autoclave at one side and to the linear actuator at the other [10]. A general scheme of this device is shown in Figure 1. In addition to the proposed layout in Figure 1, one or more Linear Variable Displacement Transducers (LVDTs) may be present in order to collect data about the gauge elongation during the test.

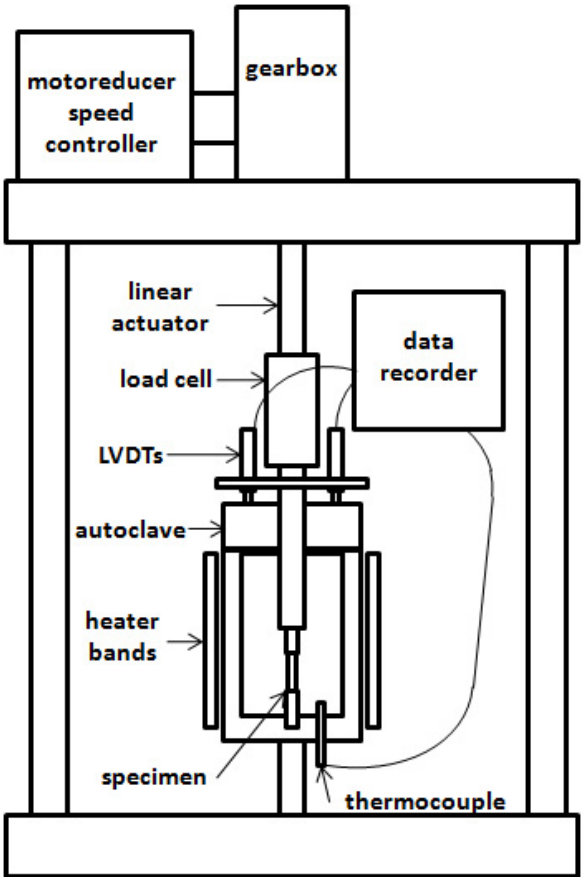

Fig. 1. Slow Strain Rate Tester scheme.

An acquisition system and a computer sample collect data which are saved as columns in a text file and separated by a mark. In this form data are pretty difficult to analyze and understand since they have not been represented or processed yet. Some laboratories may decide to develop their own tools for processing data and executing computations in order to better understand the behavior of tested materials. The TM0198 [11], by the National Association of Corrosion Engineers (NACE), points out what are the characteristic indexes which allow to compare different materials for a specific environment to select the most compatible. In this

paper the Axiomatic Design Theory is used as a reference framework for the development of software able to calculate the requested parameters and to make easier the job of technicians which have to manage and use the test data.

3. Development of the custom data post-processing software for the slow strain rate test.

3.1. Definition of the software features

The development of a custom software first requires the definition of what this software must do: the main functional requirements (FRs) of the system. In a first phase of the project, the approach for defining the main features of the software for data post processing was achieved through some classical instruments which emphasized creativity and idea generation to be filtered later. All the ideas which are drawn out through the brainstorming or the six thinking hat method [12] have been evaluated and filtered by the authors according the available time to develop the software and the list of requested indexes to be deduced from data. This list of indexes allows to select those characteristics which are more interesting from the point of view of the analysis of data from stress corrosion tests and are identified by the normative framework [10, 11]. According with these filtering criteria, and the technical regulations, the authors have used the Axiomatic Design Theory in order to develop a modular, reliable and affordable tool for the slow strain rate test post processing. In this paper the main focus is on the design stage where the structure of the software arises from the FRs it must satisfy.

3.2. Design matrices

The definition of the main features enables generation of the design matrices for the data post processing software. The design matrices give an overview of the software features from the overall requirements and solutions to the details of requested parameters in order to run the algorithms.

At this phase of development, the Axiomatic Design theory [13, 14, 15] enters strongly in the definition of functional requirements (FRs), that are what the software has to do and the design parameters (DPs) which represent how it reaches these requirements. Both FRs and DPs at lower levels are defined through the decomposition of the higher couples of FR/DP following the zigzag process. The implementation of the Axiomatic Design theory from the beginning of the project enables description of the best design for the data post processing software, avoiding less suitable solution while satisfying the user's needs. The proposed approach deals with a single FR issue where the top functional requirement is FR0: "post-process the data from slow strain rate test" and its corresponding top design parameter is DP0: "a custom software for data post-processing". The authors call this top layer level zero and go deeper within the zigzag method.

At the first level, the program for analyzing and processing data from the slow strain rate test has to provide three main functions, which are FR1: "make readable the sampled signals for stresses and strains from the test", FR2: "compute the NACE indexes" and FR3: "generate reports". At this phase the solutions which satisfy each functional requirement are DP1: "a subsystem for data importing and reading the signals", DP2: "a subsystem for executing integer math calculations" and DP3: "a subsystem for writing and printing the final report". The design matrix at this level of decomposition is provided in Table 1.

Table 1. Design Matrix for the first level of the decomposition.

	DP1	DP2	DP3
FR1	X		
FR2	X	X	
FR3	X	X	X

At this level of decomposition the matrix results in a decoupled design, i.e. these three functional requirements work as a series. This result is plausible since the calculations have to be executed after the importing of data, while the final report must be printed last, because it depends on the presence of the imported data and the calculated indexes. This is reflected in the design matrix structure since it is triangular and shows the relationships among the i-th FRs and other DPs further the i-th one. Taking into account the dependencies among the FRs, each couple of main FR/DP can be dealt like a single block, developing its own tree of lower FRs and DPs.

Each functional requirement can be further decomposed down the single algorithm, whilst the corresponding design parameter becomes the input data for the algorithm execution. Taking into account only the FR1 and the corresponding DP1, they can be further decomposed as shown below:

- FR1.1: Select the text file to be imported
- FR1.2: Load data as arrays
- FR1.3: Represent the stress and strain signals

The corresponding DPs are:

- DP1.1: The dialog box returns the name of the text file
- DP1.2: A criterion to identify single arrays
- DP1.3: A graph $\bar{\sigma} = f(\bar{\varepsilon})$

In Table 2 the mapping of the FR1 and its corresponding DP1 is shown through the second level of the decomposition. The shown matrix in Table 2 is triangular, resulting in an decoupled design. This means that all the functional requirements are mutually influenced since, in order to be accomplished, they need the outputs from the previous FR.

Table 2. Second level of decomposition for FR1/DP1

	DP1.1	DP1.2	DP1.3
FR1.1	X		
FR1.2	X	X	
FR1.3	X	X	X

The FR1.2 and FR1.3 with their corresponding DPs can be further decomposed through the zigzag mapping in order to obtain single algorithms for lower FRs and input data for the corresponding DPs:

- FR1.2.1:Load the text file content

- FR1.2.2: Recognize the arrays
- FR1.3.1: Compute the strain array $\bar{\varepsilon}$, whose the i-th element is calculated as:

$$\varepsilon_i = \frac{l_i - l_0}{l_0} \quad (1)$$

- FR1.3.2: Compute the stress array $\bar{\sigma}$, whose the i-th element is calculated as:

$$\sigma_i = \frac{4 \cdot F}{\pi \cdot d^2} \quad (2)$$

The corresponding DPs which correspond to input data are:

- DP1.2.1: The selected text file
- DP1.2.2: The column separation value (comma) that identify single arrays of data
- DP1.3.1: The elongation array $\bar{l}$ in millimetres
- DP1.3.2: The force array $\bar{F}$ in Newton

The design matrix for the third level of decomposition for FR1 and DP1 can be seen in Table 3. The moiety which deals with the decomposition of FR1.2 is triangular, while the FR1.3.1 and FR1.3.2 are linked to both their corresponding DPs and the two DP1.2.1 and DP1.2.2.

Table 3. Design Matrix for the third layer for the decomposition of FR1/DP1.

	DP1.2.1	DP1.2.2	DP1.3.1	DP1.3.2
FR1.2.1	X			
FR1.2.2	X	X		
FR1.3.1	X	X	X	
FR1.3.2	X	X		X

Considering the FR1.2.1 and FR1.2.2, the corresponding 2x2 sub-matrix is triangular, therefore FR1.2.1 and FR1.2.2 work like a series. Considering the two FR1.3.1 and FR1.3.2, the corresponding 2x4 sub-matrix is clearly rectangular with some redundancy since these functions firstly need the fulfilment of the FR.1.2. Once FR1.2 is achieved the two FR1.3.1 and FR1.3.2 can be achieved independently.

The FR2 (Compute the NACE indexes) is achieved once the content of the text file is loaded, so the outputs from the FR1 (the stress and strain arrays) are needed for achieving the FR2. The mapping of both the FR2 and its corresponding DP2 can be further developed as follow:

- FR2.1: Calculate σ_Y and ε_Y
- FR2.2: Calculate σ_{max} and ε_{max}
- FR2.3: Calculate σ_r and ε_r

While the corresponding DPs are:

- DP2.1: A subsystem to calculate σ_Y and ε_Y
- DP2.2: A subsystem to calculate σ_{max} and ε_{max}
- DP2.3: A subsystem to calculate σ_r and ε_r

The design matrix, which is shown in Table 4, is diagonal for this level of decomposition.

Table 4. Design Matrix for the second level of the decomposition for FR2/DP2.

	DP2.1	DP2.2	DP2.3
FR2.1	X		
FR2.2		X	
FR2.3			X

Each one of these subsystems to compute the yield stress and strain rather than the maximum ones or the ultimate ones can be executed apart. These three modules are pretty similar and work in the same manner, executing almost the same routines. The decomposition of these modules is thus the same and this is the reason why just one of them is dealt in the following passages. The module for calculating the yield stress and strain is going to be taken into account. The FR2.1 can be further decomposed until single algorithms are reached. Often the definition of an algorithm leads to another functional requirement which provide the input data for the first one:

- FR2.1.1: Write the value for ε_Y. In order to be accomplished, this functional requirement needs the variable "index" as input. "index" is the row in the strain array $\bar{\varepsilon}$ which contains the yield strain. In symbols the algorithm can be written as:

$$\varepsilon_Y = \bar{\varepsilon}(index; 1) \quad (3)$$

- FR2.1.2: Compute index. This functional requirement is achieved through the algorithm which counts the number of the written rows in the array which contains strain values whose the shift from the linear field is within 0,2%. In symbols this algorithm can be expressed as:

$$index = f(\bar{\varepsilon}_{el}) \quad (4)$$

- FR2.1.3: Write an array of strain values in the elastic field. This array is composed by all the data whose the difference between all the measured strain and the calculated linear equation of the elastic field is at most equal to 0,2%". This functional requirement is achieved through three inputs parameters which are the array for measured strain values $\bar{\varepsilon}$, the linear equation for approximating the elastic field and the array which contains strain values, calculated through this equation over the entire stress range $\bar{\varepsilon}_L$.
- FR2.1.4: Calculate the linear equation which approximates the elastic field. This functional requirement is achieved choosing two points with coordinates $(\varepsilon_1;\sigma_1)$ and $(\varepsilon_2;\sigma_2)$ which belong to the elastic field and calculating both the slope and the intercept of the line which best fits the elastic portion of the stress-strain curve.

The physical parameters which satisfy these functional requirements are stated below:

- DP2.1.1: The variable "index", that is the row number for the element of the strain array that is the yield value.
- DP2.1.2: The array $\bar{\varepsilon}_{el}$ which contains the values for strain whose the difference with all the values calculated through the equation for the linear field is at most equal to 0,2%.

- DP2.1.3: The coordinates of two points which belong to the elastic field.
- DP2.1.4:The imported strain array $\bar{\varepsilon}$.

The triangular design matrix for this third level of decomposition of the FR2.1/DP2.1 couple is shown in Table 5. Also in this case the single FRs need the output data from the previous ones in order to be achieved.

Table 5. Design matrix for the third level of decomposition for the FR2.1/DP2.1 couple.

	DP2.1.1	DP2.1.2	DP2.1.3A	DP2.1.4
FR2.1.1	X	X	X	X
FR2.1.2		X	X	X
FR2.1.3			X	X
FR2.1.4				X

The third main functional requirement FR3 is "Write down the final report" and it can be further decomposed in FR3.1 and FR3.2 which are:

- FR3.1: Display report
- FR3.2: Print the report

The FR3.1 is achieved through the layout of final report (a GUI, an HTML page or an Excel worksheet) which shows the obtained results, while FR3.2 is achieved through the print button (an ActiveX Control for instance), thus the two functional requirements can be reached apart the DPs are:

- DP3.1: The report layout
- DP3.2: The print button

The design matrix is diagonal for this level of decomposition and it is shown in Table 6.

Table 6. Design Matrix for the first level of decomposition for FR3/DP3

	DP3.1	DP3.2
FR3.1	X	
FR3.2		X

3.3. Software structure

The Axiomatic Design theory has a key role also in representing the software architecture through some tools like the modules and the flow chart [13], compatibly with the most common modeling strategies like the Object-Oriented techniques [16]. The flow chart is a useful and simple tool for linking the modules in order to achieve the highest FR providing the software implementation scheme. This graphic representation is based on modules that are the rows of the design matrix which yields a specific FR when they are provided with (or multiplied by) its related DPs. Each module is connected with its corresponding FR, thus, referring for example to the previous design matrices, the module M1 provides the FR1 through the DP1 while the module M2 provides FR2 through DP1 and DP2 and so on, according to the design matrix. The drawing of the chart follows the flow of data which are swapped among several block like inputs or outputs, hence the operations are executed from the innermost module to the outermost in order to operate the software.

This representation of the software architecture through the net of modules is useful also for performing a diagnosis of software failures and making easier the debugging. The software for the data post processing of slow strain rate tests should satisfy the first axiom that requires the fulfillment of each FR by just one DP which is not shared with other functions. The first axiom is satisfied at all the levels of the design hierarchy and there are no full or coupled matrix.

The software architecture is shown by the flow chart in Figure 2.

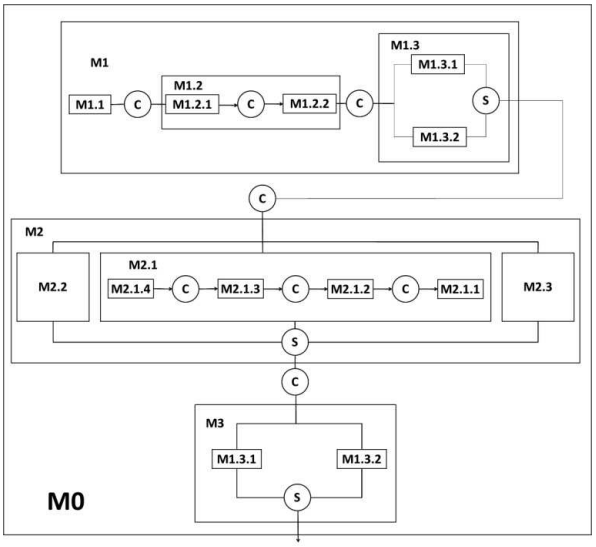

Fig. 2. Flow diagram of the overall software architecture.

According with the absence of coupled design matrices, no loop or needed feedback among blocks are present. In this case the modules often work as a sequence with their outputs which began inputs for other modules, according with the presence of triangular matrices and the corresponding decoupled design. This scheme is pretty linear and simple to carry out through the routines of the programming language, making their review easier and faster. In the flow chart, modules are represented as blocks, linked with the others through connectors to reproduce the data flow from their acquisition to the final report. The nature of the relations among several modules in different layers of the architecture is made clearer through a mark placed on the connectors, according with the scheme in [9, 10]. The mark which identifies the relation between a series of two or more modules is marked with a circled "C", meaning that the input data for the subsequent module is the output of the previous one. When two or more modules have the same input they are reciprocally independent and an "S" mark is placed on the connector between them. In this second case the group of independent modules works as a parallel. In the flow chart the module M0 links the FR0 to the DP0 and, according with the decomposition, it can be split in lower level modules M1, M2 and M3. The modules M1, M2 and M3 work as a series since

the output from M1 is required by the M2 in order to be executed. The same happens between M2 and M3. Within this three macro modules are nested other lower layer modules which may have the same data as input (parallel) or may work as a series. For example the flow chart shows that the modules M1.1, M1.2 and M1.3 work as a series, while the M2.1, M2.2 and the M2.3 work as a parallel.

4. Conclusion

A custom software for data post-analysis has been developed through the usage of the Axiomatic Design Theory. The experimental data are obtained from stress corrosion tests through the slow strain rate technique. The software executes three functions according with the NACE specifications and the user's needs, providing a reliable tool for data importing, processing and results reporting.

The Axiomatic Design enters strongly in the definition of the software architecture allowing to clarify the relationship among different modules and identify the flow of data in terms of inputs and outputs which feed the sequence of functions. A flow chart of modules is a very useful tool for software programming since it allows to set clearly the software features, optimizing the code from the first line, avoiding the trial and error approach and making a more efficient consumption of resources to be allocated for the software development. The net of modules is highly connected with the design of the software since it take into account the relations among FRs and DPs, making easier even the software debugging and revision.

The design matrices, which describe the software architecture in terms of functional requirements and corresponding design parameters for reaching them, result in decoupled or uncoupled design. This satisfies the first axiom until the single algorithms or parameters. The application of the method allows to develop a lean and agile tool for data post processing improving its efficiency and efficacy.

References

[1] European Federation of Corrosion Number 17. Corrosion resistant alloys for oil and gas production: guidance on general requirements and test methods for H_2S service. London: Maney Publishing; 2002

[2] Raymond L. Hydrogen embrittlement: prevention and control. Philadelphia: ASTM International; 1988.

[3] Qi Y, Lou H, Zheng S, Chen C, Lv Z, Xiong M. Comparison of tensile and impact behaviour of carbon steel in H_2S environments. Materials and Design 2014; 58:234-241,

[4] Woodtli J, Kieselbach R. Damage due to hydrogen embrittlement and stress corrosion cracking, Engineering Failure analysis 7, 427-450, 2000.

[5] Fallahmohammadi E, Bolzoni F, Fumagalli G, Re G, Benassi G, Lazzari L Hydrogen diffusion into three metallurgical microstructures of a C-Mn X65 and low alloy F22 sour service steel pipelines, International Journal of Hydrogen Energy 39, 13300-13313; 2014

[6] Raja VS, Shoji T. Stress corrosion cracking theory and practice. Cambridge: Woodhead Publishing; 2011.

[7] Kane RD, Greer J, Jacobs D. Stress corrosion cracking of nickel-base alloys in chloride containing environments. Corrosion 1979; 79:174.

[8] Vaught G, Greer J. High-strength nickel alloy tubulars for deep, sour gas well applications. New York: AIME Annual Meeting; 1980.

[9] Robertson W. Stress corrosion cracking and embrittlement. New York: Wiley, 154; 1956

[10] ASTM International. ASTM G129 Standard practice for slow strain rate testing to evaluate the susceptibility of metallic materials to environmentally assisted cracking. West Conshohocken, Pennsylvania; 2006

[11] Nace International. Standard test method 0198: slow strain rate test method for screening corrosion-resistant alloys (CRAs) for stress corrosion cracking in sour oilfield service. Houston; 2011.

[12] De Bono E. Six thinking hat. London: Penguin; 2009

[13] Suh NP. Axiomatic design. New York: Oxford University Press; 2001

[14] Suh NP. The principles of design. New York: Oxford University Press; 1990

[15] Thompson K. A classification of procedural errors in the definition of functional requirements in Axiomatic Design theory. The 7th International Conference on Axiomatic Design. Worcester, Massachusetts; 2013.

[16] Rumbaugh J, Blaha M, Premerlani W, Eddy F, Lorensen W. Object-Oriented Modeling and Design. New York: Prentice-Hall International; 1991