Research, Society and Development, v. 11, n. 1, e37811125132, 2022
(CC BY 4.0) | ISSN 2525-3409 | DOI: http://dx.doi.org/10.33448/rsd-v11i1.25132

Subjectivity reducing in software version criticality *classification* with the support of an expert system

Redução de subjetividade na classificação de versão de software com apoio de um sistema especialista

Reducción de la subjetividad en la clasificación de versiones de software con el apoyo de un sistema experto

Received: 12/25/2021 | Reviewed: 12/31/2021 | Accept: 01/05/2022 | Published: 01/09/2022

Dacyr Dante de Oliveira Gatto
ORCID: https://orcid.org/0000-0003-2146-4819
Universidade Nove de Julho, Brazil
E-mail: dacyr.gatto@uni9.pro.br
Renato José Sassi
ORCID: https://orcid.org/0000-0001-5276-4895
Universidade Nove de Julho, Brazil
E-mail: sassi@uni9.pro.br

Abstract
In the software version release management process, there is a need, on the part of human specialists, to classify the criticality of each software version. However, the subjectivity of this classification may be present according to the experience acquired by specialists over the years. To reduce subjectivity in the process, an Artificial Intelligence technique called Expert System (ES) can be applied to represent the knowledge of human specialists and use it in problem solving. Thus, the aim of this paper was to reduce the subjectivity in the criticality classification of the software version with the support of the Expert System. To this end, a questionnaire was developed with the objective of obtaining the criticality opinions classified as High, Medium and Low in each specialist's software version to assist in the preparation of the ES production rules. ES generated 17 production rules with a 100% confidence level applied to a production database. The results of the classification carried out by the ES corresponded to the classification carried out by the specialists in the production base, that is, the ES was able to represent their knowledge. Then, another questionnaire was applied to the specialists to verify the perception of satisfaction regarding the use of the ES with a result obtained of 4.8, considered satisfactory. It was concluded, then, that the ES supported the reduction of subjectivity in the classification of the criticality of software version.
Keywords: Subjectivity reduction; Criticality classification; Expert system; Software version release; Key performance-indicators.

Resumo
No processo de gerenciamento de liberação de versão de software, há necessidade, por parte de especialistas humanos, de classificar a criticidade de cada versão de software. No entanto, a subjetividade dessa classificação pode estar presente de acordo com a experiência adquirida por especialistas ao longo dos anos. Com o objetivo de reduzir a subjetividade no processo, uma técnica de Inteligência Artificial denominada Sistema Especialista (ES) pode ser aplicada para representar o conhecimento de especialistas humanos e utilizá-lo na resolução de problemas. Assim, o objetivo deste artigo foi reduzir a subjetividade na classificação da criticidade da versão do software com o apoio do Sistema Especialista. Para tanto, foi elaborado um questionário com o objetivo de obter as opiniões de criticidade classificadas em Alta, Média e Baixa na versão de software de cada especialista para auxiliar na elaboração das regras de produção do ES. O ES gerou 17 regras de produção com um nível de confiança de 100% aplicado a um banco de dados de produção. Os resultados da classificação realizada pelo SE corresponderam à classificação realizada pelos especialistas na base de produção, ou seja, o SE conseguiu representar os seus conhecimentos. Em seguida, outro questionário foi aplicado aos especialistas para verificar a percepção de satisfação em relação ao uso do SE com um resultado obtido de 4,8, considerado satisfatório. Concluiu-se, então, que o SE apoiou a redução da subjetividade na classificação da criticidade da versão do software.
Palavras-chave: Redução de subjetividade; Classificação de criticidade; Sistema especialista; Liberação de versão de software; Indicadores-chave de desempenho.

Resumen
En el proceso de gestión de liberación de versiones de software, existe la necesidad, por parte de especialistas humanos, de clasificar la criticidad de cada versión de software. Sin embargo, la subjetividad de esta clasificación puede estar presente según la experiencia adquirida por los especialistas a lo largo de los años. Con el fin de reducir la subjetividad en el proceso, se puede aplicar una técnica de Inteligencia Artificial llamada Specialist System (ES) para representar el conocimiento de especialistas humanos y utilizarlo en la resolución de problemas. Así, el objetivo de este artículo fue reducir la subjetividad en la clasificación de criticidad de la versión del software con el apoyo del Sistema Experto. Por lo tanto, se diseñó un cuestionario con el objetivo de obtener opiniones críticas clasificadas en Alta, Media y Baja en la versión del software de cada especialista para ayudar en la elaboración de las reglas de producción de EE. ES generó 17 reglas de producción con un nivel de confianza del 100% aplicado a una base de datos de producción. Los resultados de la clasificación realizada por la SE correspondieron a la clasificación realizada por los especialistas en la base de producción, es decir, la SE logró representar sus conocimientos. Luego, se aplicó otro cuestionario a especialistas para verificar la percepción de satisfacción con respecto al uso de la ES con un resultado obtenido de 4.8, considerado satisfactorio. Se concluyó, entonces, que la SE apoyó la reducción de la subjetividad en la clasificación de la criticidad de la versión del software.
Palabras clave: Reducción de subjetividad; Clasificación de criticidad, Sistema experto, Lanzamiento de la versión del software; Indicadores clave de rendimiento.

1. Introduction

Information Technology (IT) has assumed a strategic role within organizations, supporting organizational processes in order to allow their execution in the best possible way, increasing competitiveness in a globalized market.

In order for this competitiveness to be achieved, it seeks to avoid loss of revenue, losses to the business and to guarantee the delivery of efficient and effective IT services that support business processes. (Arrivabene, *et-al*., 2021).

In this way, IT areas have used IT Service Management (ITSM) as an instrument for managing and controlling the computing environment, providing a proactive stance to meet the needs of the organization (Axelos, 2013)

The ITSM is a set of organizational skills that promotes the integration of people, processes and technology, making them aligned with the organization's business strategy, thus providing value to organizations and the customers served by them (Barros & Salles, 2015).

Among the manageable IT services, there is the process of managing the release of hardware or software versions, which aim to build, test and deliver hardware or software services capable of supporting the specifications requested by the client, and deliver the desired results. by the organization (ITIL, 2013).

During the execution of the hardware or software version management process, key performance indicators or Key-Performance Indicators (KPIs) are generated, which help organizations to monitor the performance of the process execution, as well as to identify if there are quality in this execution (Cruz-Hinojosa & Gutiérrez-De-Mesa, 2016).

KPIs allow the development of a database with all the critical success factors of the process, being fed whenever the software version release management process is executed, thus creating a history of version releases (OGC, 2011)

With the definition of KPIs it can be seen that the software versions have different criticisms, due to the difference in complexity that each software version can present in each release (Paschek, *et al.*, 2016).

This criticality can be defined by the dependence that the components of the units and version release packages have on each other, and the consequences that the failure between these dependencies can cause to the business. In order to minimize risks in the release of software versions, it is necessary to establish methods for this purpose (Lee *et al.*, 2018).

According to Jia, *et al.* (2018) the criticality of software versions can present risks to the business and must be classified objectively for the correct treatment of these risks. With the correct treatment of these risks, it is possible to improve the analysis of the criticality of the software versions, stipulate methods to reduce this criticality and studies on the prevention of the propagation of the occurrence of failures.

During the execution of the software version release management process, the criticality of a version is classified as High, Medium and Low. The classification of criticality in categories in a textual way, such as High, Medium or Low makes the perception of this criticality instantaneous to the recipient of this information. This technique of ordering descriptors that can be suggested as synonyms indicating criticality are often used in risk matrices (Napolitano & Sassi, 2018)

This criticality, if classified incorrectly, can generate a subjective interpretation of the software version and thus, the misclassification of its criticality (Ferreira et al., 2016).

This classification is performed by specialists, executors of the process, whenever a version is made available for release, however each specialist subjectively classifies it according to their interpretation of criticality based on their knowledge from the time of experience in the function (Ferreira et al., 2016).

Each specialist interprets the criticality of the software version according to their expertise in executing the process. Its interpretation of which elements in a software version can be considered critical or does not generate divergence of opinions, so this variation can promote the imprecise classification of each software version (Monedero, *et-al.*,2008).

To reduce subjectivity in the classification of software versions in the software version release management process, an Artificial Intelligence (AI) technique can be applied, such as, for example, Expert System (ES) (monedero, *et-al.*, 2008)

According to Wagner (2017), ESs are knowledge-based systems for solving problems in a given domain, in the same way as human specialists. These knowledge-based systems are structured through a knowledge base, an inference engine, and an interface.

With the software version release management process being executed accurately, KPIs are generated to identify the critical success factors that allow to classify the criticality of the software versions. With the support of ES, subjectivity in the classification can be reduced.

Thus, allow alignment with the results expected by the organization. The aim of this paper was to reduce the subjectivity in the criticality classification of the software version with the support of the Expert System.

The study that addressed the treatment and reduction of subjectivity is considered as the main contribution of this work, which, in general, affects organizational processes by hampering decision making.

2 Bibliographic Review

2.1 Version Release Management Process

The version release management process has the purpose of identifying the criticality of a release package, or of a release unit, which is a component of a release package both in the approval environment and in the production environment, however attention should be paid to the critical subjectivity of this identification.

Identifying and classifying this criticality is necessary, and establishing methods for doing so meets this need. Heeager and Nielsen (2018) point out that due to the increased complexity of software versions, understanding and evaluating their criticality cannot be ignored.

The version release process consists of two elements: Version Release Units and Version Release Packages. These version units and packages can be released by classifying the version release type, shape and mechanism (ITIL, 2013).

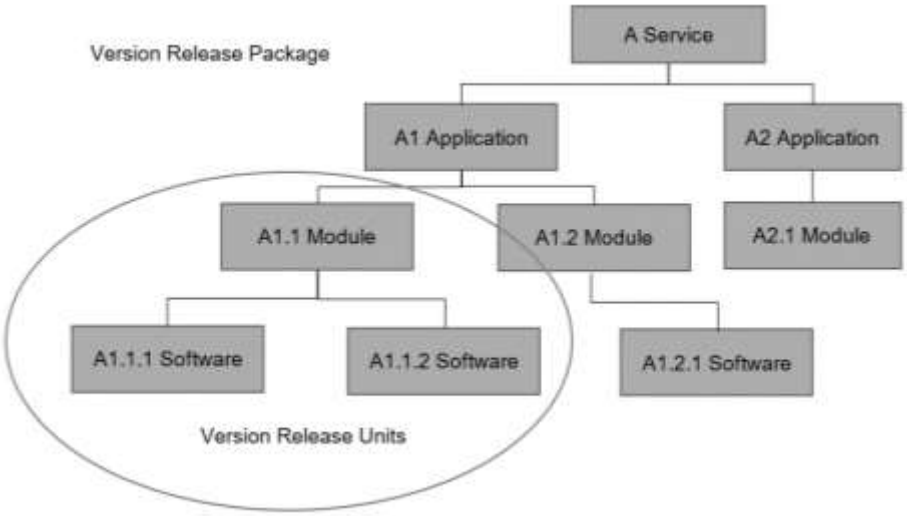
In the software version release process, the terms “Release unit” and “Release package” are defined as follows (Durkin, 1994).

a) Release Unit: small amounts of software that are released according to the need for patches or new implementations of a system, which can be just a module or an entire system.

b) Release Package: Set of release units or even a single unit, however larger. A new version of software may be released containing the new executable, object, database scripts and user manuals, for example. This set is a release package.

Figure 1 shows the graphical representation of Units and Release Packages.

Figure 1: Units and Realease Packages.



Source: Softplan (2021).

According to Figure 1, the components of a release are described as follows: a service called A when made available may consist of two applications called A1 and A2. The A1 application consists of two software modules called A1.1 and A1.2. The A1.1 module consists of two software called A1.1.1 and A1.1.2. Module A1.1 can be considered as a Software Release Package because it is composed of several smaller parts, the red ellipse highlights the Version Release Package. Software A1.1.1 can be considered a Software Release Unit because it is smaller.

A best practice in the version release management process to control the execution of the process and obtain information about the performance of the execution and its quality is the elaboration of Key Performance Indicators (KPIs) (OGC, 2011).

During the execution of the software version release management process, the following indicators can be used as KPIs (ITIL, 2013): Number of changes successfully implemented; reducing the number of unauthorized changes; reduction in the number of accumulated change requests; reduction in the number and percentage of unplanned changes and emergency corrections; reducing the number of failed changes.

KPIs can be organized in databases that allow the measurement of quality and the ability to execute the software management process. Thus, allowing to classify the criticality of a software version released by the release management process, through strategic KPIs, allowing decision making regarding the results of the execution of this process (Paschek, *et-al.*, 2016).

2.2 Expert Systems

According to Wagner (2017), Expert Systems (ESs) are based on knowledge to solve problems in a given domain in the same way that human specialists would solve.

ESs have been applied since the 1970s to solve problems in several areas. This can be seen in the work of Wagner (2017) who carried out robust research collecting 311 case studies related to the application of SEs between the years 1984 and

2016. In the research, the diversity of applications ranges from accounting services, aerospace industry, area of transport to the health area, which shows the relevance of the application in several types of problems (FARIAS *et al*., 2021).

However, SEs when applied to software have their research concentrated in the Engineering area (Rezende, *et-al.*, 2019). There are few studies that deal with the application of ESs in reducing subjectivity. The following works that deal with subjectivity can be highlighted: Lee *et-al.* (2018), development of a simulation-based test environment for security-critical software; Khan *et-al.* (2011), Knowledge-Based System Modeling for Software Process Model Selection; García-Valls *et-al.* (2018), an extensible collaborative framework for monitoring software quality in critical systems and Babar *et al.*, ES for a scalable software requirements prioritization process.

It is noteworthy that, in the bibliographic survey carried out, no work was found, which reduced subjectivity in the classification of the software version criticism with the support of an ES, which demonstrates the importance of this work.

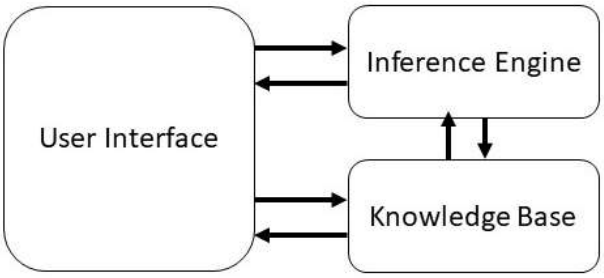
To represent the knowledge of human specialists, the ES must have not only a set of information, but also the ability to use it in solving problems. This skill represents a series of intuitive rules that the specialist uses to solve problems, and its application makes it possible, in a more economical way, to obtain acceptable, although not always optimal, solutions (PANNU, 2015).

ESs must also have the ability to learn from experience and explain what they are doing and why they are doing it, thus taking powerful training, instruction and education tools (Castelli, *et-al.*, 2017).

ESs can be classified according to the class of tasks and / or problems for which they are developed: Interpretation, Diagnosis, Monitoring, Prediction, Planning, Project, Debugging, Repair, Instruction and Control (Durkin, 1994).

The basic structure of an ES is made up of three fundamental elements: Knowledge Base, Inference Engine and User Interface. This basic structure is represented in Figure 2.

Figure 2: Basic structure of an Expert System.



Source: Waterman (1986).

The following are the three fundamental elements of the structure of an ES.

a) Knowledge Base: It can be defined as the repository for storing all the data and / or information necessary to solve a certain problem. This knowledge is classified into facts and rules, or another type of representation, such as: mathematical logic or semantic networks (Wagner, 2017).

This is not a simple collection of information. The traditional database with data, files, records and their static relationships are here replaced by a base of rules and facts and also heuristics that correspond to the knowledge of the specialist, or the specialists of the domain on which the system was built (Durkin, 1994).

Durkin (1994) explains that within the knowledge base:

- The Facts Base: It represents the knowledge that is initially known and that can be considered as a starting point for solving the problem. They are also characterized as knowledge in the public domain, easily accessible and that can be extracted through texts, manuals, standards, books, fact finding and results of experiments;

- The Rules Base: Represents the knowledge that is extracted directly from the experts. This knowledge represents the "knowledge developed by the specialist (heuristic), based on the facts already known and the deductions from them. Here, the term "heuristic" means the skill, or the simplification used by the specialist in order to optimize the search for the solution of a problem.

In this way, new knowledge can be added to the knowledge base, enabling the ES to make a decision on the problem. The representation of knowledge by rules aims to produce a general computational model of problem solving (Roldán-García, *et-al.,* 2017).

The representation of knowledge by production rules is used in ESs. The justification is the naturalness that it represents for man, because the condition-action pair to reason and decide is also used by human beings. The degree of confidence is a percentage indicating the reliability of the conclusion of the rule, which can vary from 0% to 100% (Dymova, *et-al.,* 2016)

The final structuring of the production rules is based on the IF ... THEN model, unifying the structure of the premises with the structure of the conclusion, as in the following example: SE <ATTRIBUTE 1> = <VALUE> AND <ATTRIBUTE 2> = <VALUE> THEN <ATTRIBUTE COMPLETION> = <VALUE> <DEGREE OF CONFIDENCE%>

b) Inference Machine (IM): Contains an interpreter who decides how to apply the rules to infer new knowledge, in addition to a priority list of application of these rules.

It implements ways of reasoning, techniques and search strategies, conflict resolution and treatment of uncertainty. Basically, IM performs two main functions: Interpreter / Inference: Based on the knowledge contained in the knowledge base and in the working memory, the IM determines which rules must be triggered to infer new knowledge. The IM determines or schedules the order in which the rules should be applied (Durkin, 1994).

IM is an essential element for the existence of an ES composing the core of the system. Through it, the facts, rules and heuristics that make up the knowledge base are applied in the problem-solving process.

c) User Interface: It consists of the components that allow the system to communicate with the knowledge engineer and the end user and that is easy to navigate and understand. The characteristics of the interface are directly related to the type of problem under consideration (Liao, 2005).

3. Methodology

The research methodology adopted was defined as applied research, since it aims to generate knowledge for solving problems, thus having a practical application. From the point of view of its approach, it is of a qualitative nature, whose research environment had as a direct source the data collected. Experimental research is also present, as it determines an object of study, the variables that would be able to influence it are selected, the ways of controlling and observing the effects that the variable produces on the object are defined (Yin, 2016).

The bibliographic survey was carried out in consultation with articles, books, theses, dissertations and websites related to the following terms: Specialist Systems, Software Version Release and Key-Performance Indicators.

The company Softplan offered corporate support for the development of this work by authorizing the disclosure of its name and the use of data related to the software version release process in the period between January 2017 and December 2018. The research was developed taking as based on the software version release management process for applications in the area of judicial automation, referred to as First Degree, delivered to its main customer in São Paulo with a number of users exceeding 70,000 (Softplan, 2021).

The production database was used to perform the computational experiments covering the period from January 2017 to December 2018, containing 423 records and 16 attributes. A production database is used to record the data related to the

software version release packages when they are approved and then distributed in production, this being the final environment that will use the software version.

It is worth mentioning that before the computational experiments were carried out with the production base, experiments were carried out with the homologation base, which contains 628 records and 11 attributes, that is, less in number of attributes than the production base. This experiment was carried out with the objective of verifying whether an ES could be applied on a basis with characteristics similar to the production. The results obtained were positive, but were not presented in this work because the focus was on the production base.

A homologation database is used to record the data related to the software version release package when they are released by the development team, being distributed on an homologation basis for version testing.

The hardware used to perform the computational experiments was a 2.80 GHz Intel Core i7-7700HQ Quad Core computer with 16 GB of RAM, 1TB of hard disk and 64-bit Windows 10 Pro operating system. The software used was ExSinta Version 1.1, a visual tool for creating Expert Systems (LIA, 2017).

3.1 Expert System Development and Application Phase

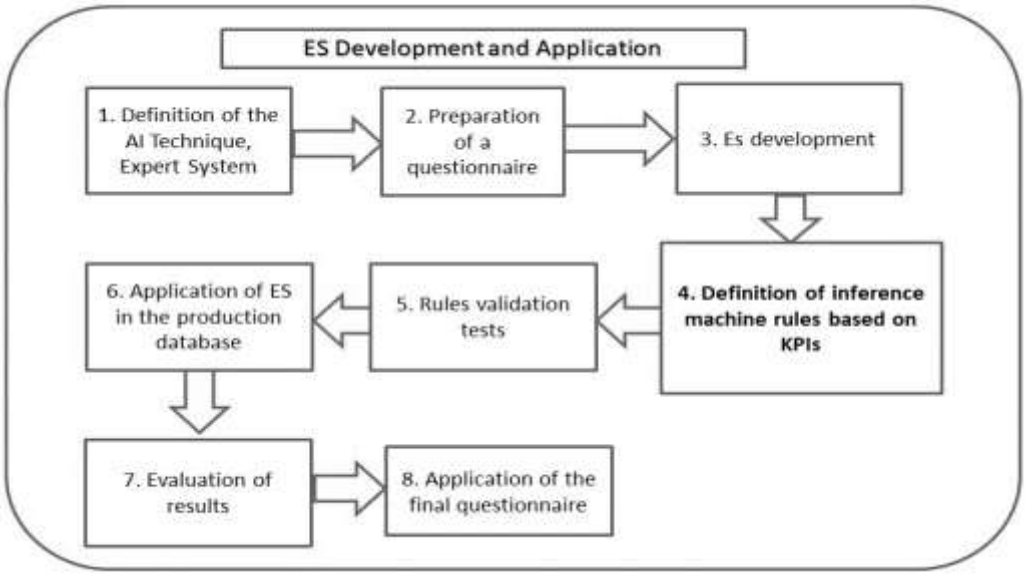
It is worth mentioning that the software version management environment was structured after the application of the following techniques and methodologies: Business Process Management (BPM), Information Technology Infrastructure Library (ITIL), Six Sigma Methodology and Project Management Body of Knowledge (PMBOK). The focus of the work is the application of ES, therefore, the application of techniques and methodologies has not been described.

The software version criticality classification is analyzed and interpreted by the specialists who execute the process, making the version criticality analysis a subjective activity, making the decision-making process more difficult. This subjectivity is due to the fact that experts have divergent opinions on the criticality of each software version. from previous experiences.

In order to reduce subjectivity in the execution of the software version release management process and to promote more accurate decision-making, ES was developed and applied.

Figure 3 shows the stages of development and application of ES

Figure 3: ES Development and application steps.



Source: Authors.

The steps presented in Figure 1 are described below:

- Step 1: the AI technique was defined, the ES as the technique to be applied to reduce subjectivity in the software version classification. For the development of the ES, it was necessary to establish a way to obtain the knowledge of the specialists. For this, each specialist was asked to assign a degree of criticality to each of the six scenarios, considering the type of release and the category of release of six execution scenarios. This assignment was called a questionnaire. Table 1 characterizes the specialist's role and his / her experience in working in the area.

Table 1: Function and length of experience in the area of professionals involved in the process.

Occupation	Time in the Area
Systems Coordinator	8 years
Systems Analyst II	5 years
Systems Analyst II	5 years
Systems Analyst I	4 years
Systems Analyst I	3 years

Source: Authors.

-Step 2: five employees of the software development company participated in the application of the questionnaire: a team coordinator and four systems analysts. The proposal for applying the questionnaire for each scenario, aimed to obtain opinions that were used for the elaboration of the ES production rules.

Table 2 shows the six scenarios of production release packages in production.

Table 2: The six production release package scenarios in production.

Version release scenarios	Items Released in Package
Server	Itens *Server*
Client	Itens *Client*
Client/Server Corretion	Itens *Client/Server* Corretion
Client/Server Implementation	Itens *Client/Server* Implementation
Client/Server PRECAT	Itens *Client/Server* PRECAT
Client/Server EST	Itens *Client/Server* EST

Source: Authors.

Exemplified in Table 3 is the sixth scenario regarding the Release Type: Client / Server (EST) and the Release Category: Correction / Implementation.

Table 3: Sixth scenario regarding the Release Type: Client / Server (EST) and the Release Category: Correction / Implementation.

6th Scenario	**Release Type: Client / Server (EST) Release Category: Patch / Implementation**		
KPIs	**Low Criticality**	**Medium Criticality**	**High Criticality**
Number of Accumulated Versions			
Number of PG5 bases updated			
Number of EST Server Objects			
Number of EST Client Objects			
Number of Extra Objects			
Number of PG5 Master Scripts			
Number of PG5 Master / Destination Scripts			
Number of EST Scripts			

Source: Authors.

With the application of the questionnaire it was obtained which values of KPIs would be considered. For the validation of the values, meetings were held with the experts in order to compare the results attributed by each specialist and, by consensus, standardize knowledge. Each scenario was called a criticality model. The results of the expert consensus for the six scenarios are presented in section 4

-Step 3: based on the final result of the application of the questionnaire after the consensus, the development of the ES began. Then, the variables to be placed in the ES and the objective variable were defined, that is, the variable that results in the classification of the software version.

-Step 4: the rules of the ES inference machine have been defined in the following format IF <ATTRIBUTE 1> = <VALUE> AND <ATTRIBUTE 2> = <VALUE> THEN <ATTRIBUTE CONCLUSION> = <VALUE> <DEGREE OF CONFIDENCE% >, based on consolidated KPIs based on expert consensus.

-Step 5: In a new meeting with the experts, the ES rules were tested and validated by the experts.

-Step 6: ES was applied to the production database.

-Step 7: for the validation of the results presented by the ES on the production base, that is, if its rules are reliable, a comparison with the classification made by the experts on the same basis was proposed (Geissman & Shultz, 1988).

-Step 8: a questionnaire with five closed questions based on the Likert Scale (Likert, 1932) was applied to the specialists in order to obtain the final perception about satisfaction in relation to the use of the ES, evaluating its interface and usability.

4. Results and Discussion

Is it presented below in the Number tables? up number? the results obtained from the experts' consensus regarding the six execution scenarios, after the application of the questionnaire.

Table 4 shows the criticality model for the Type Server software version release category Release Category Correction in Production.

Table 4: Criticality model for the Type Server software version release scenario Release Category Correction in Production.

1st Scenario	**Release Type: Server**		
	Release Category: Production Correction		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	1 till 3	4 till 6	7 or more
Number of PG5 Bases Updated	1	2 till 23	24 or more
Number of PG5 Server Objects	1 till 3	4 or 5	6 or more
Number of NET Server Objects	1 till 3	4 or 5	6 or more
Number of IND Server Objects	1 till 3	4 or 5	6 or more
Number of Extra Objects	1	2 till 4	5 or more
Number of PG5 Master Scripts	1 till 10	11 till 20	21 or more
Number of PG5 Master / Destination Scripts	1 till 10	11 till 20	21 or more
Number of NET Scripts	1 till 5	6 till 10	11 or more
Number of IND Scripts	1 till 5	6 till 10	11 or more
Number of EST Scripts	1	2 till 10	11 or more
Total Execution Time	**Till 1 Hour**	**From 1 till 3 Hours**	**Above 3 Hours**
Number of Analysts	**1**	**2**	**3 or more**

Source: Authors.

Table 5 shows the model of criticality for the scenario of software version release Type Client Release Category Correction in Production.

Table 5: Criticality model for the software version release scenario Client Type Release Category Correction in Production.

2nd Scenario	**Release Type: Client**		
	Release Category: Production Correction		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	1 till 3	4 till 6	7 or more
Number of PG5 Bases Updated	1	2 till 23	24 or more
Number of PG5 Server Objects	1	2 till 4	5 or more
Number of NET Server Objects	1	2 till 4	5 or more
Number of IND Server Objects	1	2 till 4	5 or more
Number of Extra Objects	1	2 till 4	5 or more
Number of PG5 Master Scripts	1	2 till 3	4 or more
Number of PG5 Master / Destination Scripts	1 till 10	11 till 20	21 or more
Number of NET Scripts	1 till 10	11 till 20	21 or more
Number of IND Scripts	1 till 5	6 till 10	11 or more
Number of EST Scripts	1 till 5	6 till 10	11 or more
Total Execution Time	1 till 5	6 till 10	11 or more
Number of Analysts	**Till 2 Hous**	**From 2 till 4 hours**	**Above 4 Hours**
Number of Accumulated Versions	**1**	**2**	**3 or more**

Source: Authors.

Table 6 shows the criticality model for the software version release scenario Type Client / Server Release Category Correction in Production.

Table 6: Criticality model for the Client / Server type software release scenario (Standard Package) Release category Correction in Production.

3nd Scenario	**Release Type: Client / Server (Standard package)**		
	Release Category: Production Correction		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	1 till 3	4 till 6	7 or more
Number of Bases PG5 Atualizadas	1	2 till 23	24 or more
Number of Objects PG5 *Server*	1 till 3	4 ou 5	6 or more
Number of Objects PG5 *Client*	1 ou 2	3 ou 4	5 or more
Number of Objects NET *Server*	1 till 3	4 ou 5	6 or more
Number of Objects NET *Client*	1	2 till 4	5 or more
Number of Objects IND *Server*	1 till 3	4 ou 5	6 or more
Number of Objects IND *Client*	1 ou 2	3 ou 4	5 or more
Number of Objects PSS *Client*	1	2 till 4	5 or more
Number of Objects Extras	1	2 till 4	5 or more
Number of Scripts PG5 *Master*	1 till 5	6 till 20	21 or more
Number of Scripts PG5 *Master*/Destination	1 ou 2	3 till 20	21 or more
Number of Scripts NET	1 till 5	6 till 20	21 or more
Number of Scripts IND	1 till 5	6 till 20	21 or more
Number of Scripts EST	1	2 till 20	21 or more
Total Execution Time	**Till 2 Hours**	**From 2 till 4 hours**	**Above 4 Hours**
Number of Analysts	**1**	**2**	**3 or more**

Source: Authors.

Table 7 shows the model of criticality for the Scenario of software version release Type Client / Server Release Category Implementation in production.

Table 7: Criticality model for the Software Version Release Scenario Client / Server Type Release Category Production Implementation.

4nd Scenario	**Release Type: Client / Server**		
	Release Category: Production Implementation		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	1	2 till 4	5 or more
Number of Bases PG5 Atualizadas	1	2 ou 3	4 or more
Number of Objects PG5 *Server*	1 till 3	4	5 or more
Number of Objects PG5 *Client*	1 till 3	4	5 or more
Number of Objects NET *Server*	1 till 3	4	5 or more
Number of Objects NET *Client*	1 till 3	4	5 or more
Number of Objects IND *Server*	1 till 3	4	5 or more
Number of Objects IND *Client*	1 till 3	4	5 or more
Number of Objects PSS *Client*	1	2	3 or more
Number of Objects Extras	1	2 till 4	5 or more
Number of Scripts PG5 *Master*	1 till 5	6 till 20	21 or more
Number of Scripts PG5 *Master*/Destination	1 till 10	11 till 20	21 or more
Number of Scripts NET	1 till 5	6 till 20	21 or more
Number of Scripts IND	1 till 5	6 till 20	21 or more
Number of Scripts EST	1 till 5	6 till 20	21 or more
Total Execution Time	**Till 3 Hours**	**From 3 till 6 Hours**	**Above 6 Hours**
Number of Analysts	**2**	**3**	**4 or more**

Source: Authors.

Table 8 shows the model of criticality for the Scenario of software version release Type Client / Server PRECAT Release Category Correction in production.

Table 8: Model of criticality for the Scenario of software version release Type Client / Server PRECAT Release Category Correction in Production

5nd Scenario	**Release Type: Client / Server (PRECAT)**		
	Release Category: Production Correction		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	1	2 till 4	5 or more
Number of Bases PG5 Atualizadas	1	2 till 23	24 or more
Number of Objects PG5 *Server*	1 till 3	4	5 or more
Number of Objects PG5 *Client*	1	2	3 or more
Number of Objects NET *Server*	0	0	0
Number of Objects NET *Client*	0	0	0
Number of Objects IND *Server*	0	0	0
Number of Objects IND *Client*	0	0	0
Number of Objects PSS *Client*	0	0	0
Number of Objects Extras	1	2	3 or more
Number of Scripts PG5 *Master*	1 till 5	6 till 20	21 or more
Number of Scripts PG5 *Master*/Destination	1 till 5	6 till 20	21 or more
Number of Scripts NET	0	0	0
Number of Scripts IND	0	0	0
Number of Scripts EST	0	0	0
Total Execution Time	Till 1 Hour	De 1 a 2 Hours	Acima 2 hours
Number of Analysts	1	2	3 or more

Source: Authors.

Table 9 shows the model of criticality for the Scenario of software version release Type Client / Server EST Release Category Correction / Implementation in production.

Table 9: Criticality model for the Software Version Release Scenario Type Client / Server EST Release Category Correction / Implementation in Production.

6nd Scenario	**Release Type: Client / Server (EST)**		
	Release Category: Production Correction/Implementation		
	Low Criticality	Medium Criticality	High Criticality
Number of Accumulated Versions	-	1	2 or more
Number of Bases PG5 Atualizadas	-	1	2 or more
Number of Objects EST *Server*	-	1 till 3	4 or more
Number of Objects EST *Client*	-	1	2 or more
Number of Objects Extras	-	1 ou 2	3 or more
Number of Scripts PG5 *Master*	-	1 till 5	6 or more
Number of Scripts PG5 *Master*/Destination	-	1 till 5	6 or more
Number of Scripts EST	-	1 till 5	6 or more
Total Execution Time	-	Till 1 Hour	Above 2 Hours
Number of Analysts	-	1	2 or more

Source: Authors.

The number of production rules generated by the ES was seventeen (17). Table 10 shows an example of a rule and its characterization.

Table 10: Example of characterization of a production rule generated by the ES for the production base.

Rule 1	**Characterization**
ES Release Type = Server	Defines within which Scenario the inference will be made
AND 1_Number of Accumulated Versions = 1 till 3	Defines the Number of Versions to be released
AND 1_Number of PG5 Bases Updated = 1	Define the Number of PG5 Bases that will be updated
OR 1_Number of PG5 Bases Updated = 2 till 23	
OR 1_Number of PG5 Bases Updated = 24 or more	
AND 1_Number of Objects PG5 Server = 1 till 3	Defines the Number of Objects PG5 to be distributed
AND 1_Number of Objects NET Server = 1 till 3	Defines the Number of Objects NET to be distributed
AND 1_Number of Objects IND Server = 1 till 3	Defines the Number of Objects IND to be distributed
AND 1_Number of Objects Extras = 1	Defines the Number of Objects Extras to be distributed
AND 1_Number of Scripts PG5 Master = 1 till 10	Defines the Number of PG5 Master Scripts to be executed
AND 1_Number of Scripts PG5 Master / Destination = 1 till 10	Defines the Number of PG5 Master / Destination Scripts to be executed
AND 1_Number of Scripts NET = 1 till 5	Defines the Number of NET Scripts to be executed
AND 1_Number of Scripts IND = 1 till 5	Defines the Number of IND Scripts to be executed
AND 1_Number of Scripts EST = 1	Defines the Number of EST Scripts to be executed
AND 1_Total Execution Time	Sets the Total Execution Time of the version release
AND 1_Number of Analysts	Defines the Number of Analysts executing the release
THEN Version Criticality = Low Criticality CNF 100%	Classifies the software version and its degree of confidence

Source: Authors.

Three examples of production rules generated by the ES are presented below.

Rule 2

IF Release Type = Client
AND 2_Number of Accumulated Versions = 1 till 3
ND 2_Number of PG5 Bases Updated = 1
OR 2_Number of PG5 Bases Updated = 2 till 23
OR 2_Number of PG5 Bases Updated = 24 or more
AND 2_Number of Objects PG5 Client = 1
AND 2_Number of Objects NET Client = 1
AND 2_Number of Objects IND Client = 1
AND 2_Number of Objects Extras = 1
AND 2_Number of Objects PSS Client = 1
AND 2_Number of Scripts PG5 Master = 1 till 10
AND 2_Number of Scripts PG5 Master / Destination = 1 till 10
AND 2_Number of Scripts NET = 1 till 5
AND 2_Number of Scripts IND = 1 till 5

AND 2_Number of Scripts EST = 1 till 5
AND 2_Total Execution Time = Till 2 Hours
AND 2_Number of Analysts = 1
THEN Version Criticality = Low Criticality (Degree of Confidence = 100%)

Rule 3

IF Release Type = Client / Server Correction
AND 3_Number of Accumulated Versions = 1 till 3
AND 3_Number of PG5 Bases Updated = 1
OR 3_Number of PG5 Bases Updated = 2 till 23
OR 3_Number of PG5 Bases Updated = 24 or more
AND 3_Number of Objects PG5 Server = 1 till 3
AND 3_Number of Objects PG5 Client = 1 or 2
AND 3_Number of Objects NET Server = 1 till 3
AND 3_Number of Objects NET Client = 1
AND 3_Number of Objects IND Server = 1 till 3
AND 3_Number of Objects IND Client = 1 or 2
AND 3_Number of Objects PSS Client = 1
AND 3_Number of Objects Extras = 1
AND 3_Number of Scripts PG5 Master = 1 till 5
AND 3_Number of Scripts PG5 Master / Destination = 1 or 2
AND 3_Number of Scripts NET = 1 till 5
AND 3_Number of Scripts IND = 1 till 5
AND 3_Number of Scripts EST = 1
AND 3_Total Execution Time = Till 2 Hours
And 3_Number of Analysts = 1
THEN Version Criticality = Low Criticality (Degree of Confidence = 100%)

Rule 4

IF Release Type = Client / Server Implementation
AND 4_Number of Accumulated Versions = 1
AND 4_Number of PG5 Bases Updated = 1
OR 4_Number of PG5 Bases Updated = 2 or 3
OR 4_Number of PG5 Bases Updated = 4 or more
AND 4_Number of Objects PG5 Server = 1 till 3
AND 4_Number of Objects PG5 Client = 1 till 3
AND 4_Number of Objects NET Server = 1 till 3
AND 4_Number of Objects NET Client = 1 till 3
AND 4_Number of Objects IND Server = 1 till 3
AND 4_Number of Objects IND Client = 1 till 3
AND 4_Number of Objects PSS Client = 1
AND 4_Number of Objects Extras = 1
AND 4_Number of Scripts PG5 Master = 1 till 5
AND 4_Number of Scripts PG5 Master / Destination = 1 till 10
AND 4_Number of Scripts NET = 1 till 5
AND 4_Number of Scripts IND = 1 till 5
AND 4_Number of Scripts EST = 1 till 5
AND 4_Total Execution Time = Till 3 Hours
AND 4_Number of Analysts = 2
THEN Version Criticality = Low Criticality (Degree of Confidence = 100%)

Table 11 presents an example of the Production database for the period from January 2017 to December 2018 containing the rating of criticality performed by the human specialist before the consensus, the rating of criticality performed by the human specialist after the consensus and the criticality classification carried out by the ES for the purpose of comparison.

Table 11: Production Database from January 2017 to December 2018 used in the comparison between the specialist and the ES.

Date	Version	Type	Complexity	Number of Versions	Number of Objects	Total Number of Scripts	Total time (Hours)	Quantity of Analysts	Human Specialist Classification Before Consensus	Human Specialist Classification After Consensus	ES classification
17/jan	1.8.15-23_D	SERVER	CORRECTION	1	5	20	01:20	1	MEDIUM	MEDIUM	MEDIUM
19/jan	1.8.15-23_E	SERVER	CORRECTION	1	5	20	02:05	1	MEDIUM	MEDIUM	MEDIUM
10/jan	1.8.29-0(EST)	CLIENT/ SERVER	CORRECTION	1	5	46	01:50	1	MEDIUM	MEDIUM	MEDIUM
16/jan	1.8.16-9 (EST)	CLIENT/ SERVER	CORRECTION	1	5	1	00:45	1	MEDIUM	MEDIUM	MEDIUM
21/jan	1.8.28-27_A	CLIENT/ SERVER	CORRECTION	8	21	732	05:05	2	MEDIUM	MEDIUM	MEDIUM
27/jan	1.8.28-30_A (PRECAT)	CLIENT/ SERVER	CORRECTION	59	7	496	09:00	1	MEDIUM	MEDIUM	MEDIUM
28/feb	1.8.15-32_B	SERVER	CORRECTION	1	5	20	00:20	1	MEDIUM	LOW	LOW
02/mar	1.8.15-32_C	SERVER	CORRECTION	1	5	20	00:30	1	MEDIUM	LOW	LOW
28/jan	1.8.28-30_A	CLIENT/ SERVER	CORRECTION	3	7	202	03:10	2	MEDIUM	MEDIUM	MEDIUM
16/mar	1.8.15-35_C	SERVER	CORRECTION	1	5	20	00:30	1	MEDIUM	LOW	LOW
13/mar	1.8.15-35_B	CLIENT/ SERVER	CORRECTION	6	23	290	01:25	2	MEDIUM	MEDIUM	MEDIUM
17/mar	1.8.20-4_B (FOF/EF)	CLIENT/ SERVER	IMPLEMENTATION	15	40	608	06:40	2	MEDIUM	HIGH	HIGH
20/mar	1.8.20-4_C (FOF/EF)	CLIENT/ SERVER	CORRECTION	1	7	2	01:40	1	LOW	LOW	LOW
21/mar	1.8.20-4_D (FOF/EF)	CLIENT/ SERVER	CORRECTION	1	7	2	00:40	1	LOW	LOW	LOW
22/mar	1.8.20-4_E (FOF/EF)	CLIENT/ SERVER	CORRECTION	1	7	2	01:20	1	LOW	LOW	LOW
23/mar	1.8.20-4_F (FOF/EF)	CLIENT/ SERVER	CORRECTION	1	7	2	01:40	1	LOW	LOW	LOW
24/mar	1.8.20-4_G	CLIENT/ SERVER	IMPLEMENTATION	1	25	5211	03:40	2	ALTA	ALTA	ALTA
27/mar	1.8.20-4_H	CLIENT/ SERVER	CORRECTION	1	13	22	01:40	1	LOW	LOW	LOW
29/mar	1.8.20-4_I	CLIENT/ SERVER	CORRECTION	1	7	20	00:40	1	LOW	MEDIUM	MEDIUM
30/mar	1.8.20-4_J	CLIENT/ SERVER	CORRECTION	1	7	20	01:05	1	LOW	MEDIUM	MEDIUM
01/abr	1.8.20-4_L	CLIENT	CORRECTION	1	2	20	01:30	1	MEDIUM	MEDIUM	MEDIUM
31/mar	1.8.20-4_K	CLIENT/ SERVER	CORRECTION	1	9	40	00:45	1	LOW	MEDIUM	MEDIUM
05/oct	1.8.23-13	CLIENT/ SERVER	CORRECTION	3	15	345	01:55	2	MEDIUM	MEDIUM	MEDIUM
06/oct	1.8.23-13_A	CLIENT/ SERVER	CORRECTION	1	7	20	00:45	2	LOW	MEDIUM	MEDIUM
24/apr	1.8.20-7_F/1.8.22-1	PRO/EST	CORRECTION	2	9	138	00:30	1	MEDIUM	MEDIUM	MEDIUM
27/apr	1.8.20-7_G	SERVER	CORRECTION	1	5	20	00:40	1	MEDIUM	LOW	LOW
09/oct	1.8.23-13_B (PRECAT)	CLIENT/ SERVER	CORRECTION	1	7	1	00:30	1	LOW	LOW	LOW

Source: Authors.

In relation to Table 11, the divergent classifications among human specialists before and after the consensus and the classification carried out by the ES stood out in yellow. Based on the results in Table 14, it was observed that of the 275 classifications performed by the specialists, 62 presented divergences in the classification performed by the specialists before the consensus and after the consensus. It was also observed that the classification issued by the ES corresponded to the classification of the specialists after the consensus.

The results are presented in Table 12, considering the equalities and divergences in the criticality classification of the software version, between the specialists before consensus and the ES, in the period from January 2017 to December 2018.

Table 12: Consolidated Results of the Version Criticality Classification in Scenarios from January 2017 to December 2018.

Scenarios	Success	Disagreements	Percentage success	Divergent Percentage
CLIENT	5	0	100%	0%
CLIENT / SERVER CORRECTION	96	29	76,8%	23,2%
CLIENT/SERVER IMPLEMENTATION	10	7	58,8%	41,2%
EST	12	2	85,7%	14,3%
PRECAT	79	6	92,9%	7,1%
SERVER	12	17	41,4%	58,6%

Source: Authors.

Evaluating the results of the divergences in the software version criticality classification with the production database, shown in Table 15, it was observed that the Scenarios Client / Server Implementation and Server presented an MEDIUM of 50% divergence in the classification among experts before consensus and in ES classification.

These divergences characterize the subjectivity in the analysis of each version by the specialists, who even with simpler release packages had different classifications before the consensus in relation to the ES.

Regarding the comparison between the results of the experts 'classification after the consensus with the results obtained by the ES, it can be observed that the ES continued to present classifications that reflected the experts' knowledge after the consensus.

In Step 8, a questionnaire was applied to specialists in order to obtain the final perception about satisfaction in relation to the use of the ES, evaluating its interface and usability, as well as the results of the criticality rating of the software version obtaineds.

Table 13 presents the results of the questionnaire on the satisfaction of using the ES in the production base.

Table 13: Results of the ES use satisfaction questionnaire.

Question	answer 1	answer 2	answer 3	answer 4	Answer 5
Did the ES interface make the objective of the proposed questioning clear?	5	5	5	5	4
During the experiments, was it easy to understand the usability and navigation between the ES screens?	5	5	4	5	4
During the experiments with the Production Database, did the ES results meet your expectations?	5	5	5	5	5
At the end of the experiments with the ES, did your perception on the use and results obtained demonstrate a reduction in subjectivity in the classification of software version?	5	5	5	5	5

Source: Authors.

Analyzing the results of the questionnaire, the following expert opinion was obtained, as shown in Table 14.

Table 14: Consolidated result of the ES use satisfaction questionnaire.

Questioning	Average of Each Item
Interface	4,8
Usability	4,4
Experience with the Production Database	5
Regarding the results of reducing subjectivity in the classification of software version	5
AVERAGE Total	4,8

Source: Authors.

Based on the results presented in Table 18, it was observed that the evaluation of the ES by the specialists met the expectations regarding the interface, the use, and the reduction of subjectivity in the classification of software version.

5. Conclusion

To reduce subjectivity in the opinion of human experts regarding the criticality of the software version, the development and application of an ES was proposed. Thus, when applying ES in the classification of software version in the production database, the results obtained were consistent with the classification after the consensus made by human specialists on criticality. There was a reduction in subjectivity in the process.

The results of the software version classification issued by ES were validated when compared with the classification results after consensus by human specialists. It is concluded that the general objective was achieved by reducing subjectivity in the criticality classification of software version with the support of ES.

The study that addressed the treatment and reduction of subjectivity is considered as the main contribution of this work, which, in general, affects organizational processes by hampering decision making.

In the academic field, this research contributed to the academic literature on the subjectivity reduction subject, using an AI technique in the software version release management process.

At the corporate level, this research shows the practical application of methodologies and good practices in process management, IT and projects applied in an aligned and integrated manner, together with the application of an AI technique, considered in this scenario as an emerging technology, and a differential for more objective decision-making processes.

In the social sphere, the fact that this research was carried out in the corporate environment of a software development company, which provides services to customers who effectively work with social welfare, provided a maturity in the execution of the version release management process. software, reducing the unavailability of the systems used for this purpose. Consequently, the end users of the applications of this client of the software development company have benefited from the decrease in the unavailability of the systems, which play an important role in the provision of judicial services to society.

The application of another AI technique, Case Based Reasoning (CBR), is indicated as a continuation of the research when considering the release package scenarios as cases on a case basis that would be updated automatically.

References

Arrivabene, A., Sassi, R. J., Andrelo, P. F. A., Moura, M. L. A. De O (2021). Analysis of the impact of adequacy on operational information technology processes to the requirements of the Sarbanes-Oxley act in a financial company. *Research, Society and Development*, 10(1), e7710111374. 10.33448/rsd-v10i1.11374.

Axelos. (2013) Global best practice. ITIL Maturity Model and Self-Assessment Service: User Guide. Axelos Limited. http://www.axelos.com.

Babar, M. I., Ghazali, M., Jawawi, D. N. A., Shamsuddin, S. M., & Ibrahim, N. (2015). PHandler: An expert system for a scalable software requirements prioritization process. *Knowledge-Based Systems*, 84, 179-202. https://doi.org/10.1016/j.knosys.2015.04.010

Barros, M. D., & Salles, C. A. L. (2015). Mapping of the Scientific production on the Itil application published in the national and international literature, *Procedia Computer Science*, 55, 102-111. 10.1016/j.procs.2015.07.013

Castelli, M., Manzoni, L., Vanneschi, L., & Popovič, A. (2017). An Expert System for Extracting Knowledge From Customers' Reviews: The Case of Amazon. com, inc. *Expert Systems With Applications*, 84, 117-126. https://doi.org/10.1016/j.eswa.2017.05.008.

Cruz-Hinojosa, N. J., & Gutiérrez-De-Mesa, J. A. (2016). Literature review of the situation research faces in the application of ITIL in Small and Medium Enterprises. *Computer Standards & Interfaces*, 48, 124-138. doi.org/10.1016/j.csi.2016.05.001

Napolitano, D. M. R.; & Sassi, R. J. (2018). Um Sistema de Inferência Fuzzy para Análise de Riscos em Projetos Baseado em Matrizes de Probabilidade e Impacto. *NAVUS Revista de Gestão e Tecnologia*, 8, 69-89.

Durkin, J. (1994) Expert systems: design and development. Macmillan Publishing.

Dymova, L., Sevastjanov, P., & Kaczmarek, K. (2016). A Forex Trading Expert System Based on a New Approach to The Rule-base Evidential Reasoning. *Expert Systems With Applications*, 51, 1–13. doi.org/10.1016/j.eswa.2015.12.028.

Farias, E. B. P., Gatto, D. D. O., Romero, M., Moura, M. L. A. O., & Sassi. R. (2021). Processo de Enfermagem Apoiad por sistema Especialista na Aplicação das Escalas de Glasgow e Braden em um Hospital Público Brasileiro. *NAVUS Revista de Gestão e Tecnologia*, 11, 114-129.

Ferreira, C., Nery, A., & Pinheiro, P. C. (2016). A Multi-Criteria Model in Information Technology Infrastructure Problems, *Procedia Computer Science*, 91, 642-651. 10.1016/j.procs.2016.07.161.

García-Valls, M.; Escribano-Barreno, J.; Munoz, J. G. (2018). An extensible collaborative framework for monitoring software quality in critical systems. *Information and Software Technology*, v. 107, p. 3-17. 10.1016/j.infsof.2018.10.005.

Geissman, J. R., & Shultz, R. D. (1988). Verification and validation of expert systems. AI Expert, San Francisco, 1(1), 26-33.

Heeager, L. T., & Nielsen, P. A. (2018). A Conceptual Model Of Agile Software Development In A Safety-Critical Context: A Systematic Literature Review. *Information And Software Technology*, 103, 22-39. 10.1016/J.Infsof.2018.06.004.

ITIL. (2013) ITIL Service Lifecycle Publication Suite. Editora Tso, Edição: Uk Ed.

Jia, G., Ming, Y., Bowen, Z., Yuxin, Z., Jun, Y., & Xinyu, D. (2018). Nuclear Safety-Critical Digital Instrumentation And Control System Software: Reliability Demonstration. *Annals Of Nuclear Energy*, 120, 516-527. 10.1016/J.Anucene.2018.06.003.

Khan, A. R.; Rehman, Z. U.; & Amim, H. U. (2011). Knowledge-Based System's Modeling for Software Process Model Selection. *International Journal of Advanced Computer Science and Applications* (IJACSA), 2(2), 20-25. https://doi.org/10.14569/IJACSA.2011.020205

Lee, S. H., Lee, S. L, Park, J., Lee, E., & Kang, H. G. (2018). Development Of Simulation-Based Testing Environment For Safety-Critical Software. *Nuclear Engineering And Technology*, 50, 570-581. 10.1016/J.Net.2018.02.007.

LIA. (2017). ExSinta Versão 1.1 Uma Ferramenta Visual Para Criação De Sistemas Especialistas Manual Do Usuário. Laboratório De Inteligência Artificial. de: <Http://Www.Lia.Ufc.Br>.

Liao, S. (2005). Expert System Methodologies And Applications – A Decade Review From 1995 To 2004. *Expert Systems With Applications*. 28, 93-103. Https://Doi.Org/10.1016/J.Eswa.2004.08.003.

Monedero, I., León, C, Denda, R., & Luque, J. (2008). Datacab: A Geographical-Information System Based Expert System For The Design Of Cable Networks. Expert Systems, 25(4), 335–348. Doi.Org/10.1111/J.1468-0394.2008.00445.X.

OGC. (2011). Office Of Government Commerce. ITIL - Service Strategy, Norwich: TSO Information & Publishing Solutions, 2011.

Pannu, A. (2015). Survey On Expert System And Its Research Areas. *International Journal Of Engineering And Innovative Technology (IJEIT*), 4 (10), 104-108, 2015.

Paschek, D., Rennung, F., Trusculescu, A., & Draghici, A. (2016). Corporate Development With Agile Business Process Modeling As A Key Success Factor. *Procedia Computer Science*, 100, 1168-1175. Https://Doi.Org/10.1016/J.Procs.2016.09.273.

Rezende, A. V.; Silva, A.; Britto, A.; & Amaral, R. (2019). Software project scheduling problem in the context of search-based software engineering: A systematic review. *Journal of Systems and Software*. 155, 43-56.

Roldán-García, M. D. M., García-Nieto, J., & Aldana-Montes, J. F. (2017). Enhancing Semantic Consistency In Anti-Fraud Rule-Based Expert Systems. *Expert Systems With Applications*, Málaga, Spain, 90, 332-343. Http://Dx.Doi.Org/10.1016/J.Eswa.2017.08.036.

Softplan. "Quem Somos" (2021). Disponível Em: http://www.softplan.com.br/a-softplan/quem-somos/. Acesso Em 21/12/2021.

Wagner, W. P. (2017). Trends In Expert System Development: A Longitudinal Content Analysis Of Over Thirty Years Of Expert System Case Studies. *Expert Systems With Applications*, 76, 85-96. 10.1016/J.Eswa.2017.01.028.

Yin, R. K. (2016). Pesquisa Qualitativa Do Início Ao Fim. (2a ed.).