UMA FERRAMENTA MATEMÁTICA PARA ANÁLISE CRÍTICA DO CICLO DE VIDA DE UM SOFTWARE: METODOLOGIA 2E2S

Domingos José Ribeiro *****

Wagner Costa Botelho **

Renata Maciel Botelho ***

RESUMO

A gestão da qualidade consiste em uma estratégia orientada, que visa a organização sistemática dos processos em uma instituição. Na engenharia da computação, este conceito objetiva garantir a qualidade do *software* através da definição e normatização de processos de desenvolvimento. Desta forma, grande parcela das metodologias de qualidade existentes tem seu cerne voltado à avaliação de processos, e são utilizadas como forma de encontrar possíveis defasagens e oportunidades de melhoria. No entanto, esse trabalho busca elucidar uma diferente perspectiva para a análise qualitativa deste ramo de atividade onde, além de aplicação dos principais fundamentos da matemática, da qualidade no ciclo de vida do *software*, também busca alinhar as melhores práticas da segurança da informação para a sustentabilidade dos negócios caso haja algum tipo de falha, independentemente de onde possa ocorrer. Esta nova estrutura analítica visa encaminhar o *software* para a excelência, de maneira continua, mesmo que sejam atingidos todos os quesitos da metodologia, sendo necessária a recorrência da aplicação das etapas propostas sobre o software para a garantia da continuidade dos negócios.

Palavras chave: *Software*, Eficiência e Eficácia, Satisfação do Cliente, Segurança da Informação, Modelo Matemático

1. Introdução

Os autores dessse trabalho acompanharam o desenvolvimento tecnológico desde o início de suas trajetórias profissional, ou seja, desde a criação da internet, desde a época que a rede de informações ainda embrionária era acessada através das "BBSs" por telefone discado conectado a um modem analógico com transmissão assíncrona de dados a uma velocidade de 1.200 bits por segundo.

Ao longo das últimas três décadas, pela vivência profissional e acadêmica dos autores dessa pesquisa (Engenharia da Qualidade, Engenharia de Produção e Matemática) pode ser observado que o panorama geral do setor apresenta situações críticas de falhas que tem potencial para comprometer a continuidade de negócios empresariais após incidentes técnicos.

Por conta dessas falhas, foi percebido que os indicadores matemáticos que medem a qualidade de um *software* são insuficientes para uma análise estruturação matemática mais profunda, a respeito de suas causas e efeitos, bem como os respectivos métodos de correção.

Em decorrência disto, esse trabalho apresenta uma nova metodologia para análise crítica de ciclo de vida de um *software*, abordada mais amplamente, onde serão associados os processos primordiais do Planejamento Avançado da Qualidade do Produto (APQP, 2016) e os padrões internacionais de desenvolvimento de *software* (ISO 12207 e 25010). Estes, por sua

*Master em Engenharia da Qualidade (POLI-USP) ribeiro_domingos@hotmail.com
**Doutor em Engenharia de Produção (UNIP) wagner_botelho@terra.com.br
***Mestranda em Educação Matemática (UNIBAN) renatabotelho@hotmail.com

vez, são categorizados nos pilares: Eficiência, Eficácia, Satisfação do cliente e Segurança (2E2S), com foco na segurança da informação, modelagem matemática, sustentabilidade e no planejamento da continuidade dos negócios.

Baseado na premissa de que as normas, genericamente, visam estabelecer uma efetiva comunicação entre os setores envolvidos no planejamento e desenvolvimento de um produto, estes princípios serão direcionados para a análise processual na composição de um *software*, buscando relacionar todas as etapas do processo aos fundamentos primordiais definidos por meio da teoria 2E2S, como controle de deadlines e cronogramas, buscando a redução ao mínimo da possibilidade de existência de modos de falha, além da minimização dos riscos pertinentes a este processo.

Desta forma, será analisado o ciclo de vida de um *software*, através do ponto de vista da melhoria continuada, com o propósito principal de estabelecer uma relação "ganha-ganha" entre desenvolvedor e empresa, através da relação inversamente proporcional entre a utilização de recursos e a obtenção de resultados. Para isso, serão utilizadas as principais fases do APQP, que compreendem o planejamento, projeto, verificação, validação, feedback e ações corretivas, associando-as às principais fases de desenvolvimento de *software*, de forma a realizar uma análise quantitativa destes conceitos através de uma metodologia de desenvolvimento autoral, considerando os itens mais importantes para a indicação de pontuação processual ao longo do ciclo de vida do *software*, com foco na segurança, sustentabilidade e continuidade dos negócios.

2. Referencial teórico

O APQP - sigla para o termo em inglês *Advanced Product Quality Planning* ou Planejamento Avançado da Qualidade do Produto – trata de uma série de procedimentos e técnicas usadas para gerenciar a qualidade durante a cadeia produtiva. Estes procedimentos foram desenvolvidos e padronizados pela AIAG (*Automotive Industry Action Group*, 2008), formado por três das empresas de maior reconhecimento do ramo automotivo – mais conhecidas como *Big Three*: General Motors, Ford, Chrysler e seus fornecedores.

Atualmente, a metodologia estabelecida pelo APQP (figura 1) é utilizada em empresas de diversos portes, especialmente no ramo automotivo, a fim de assegurar a qualidade dos produtos e processos desenvolvidos em sua planta, e é regida pelo manual de referência, atualmente em sua segunda edição, publicado no Brasil pelo IQA – Instituto de Qualidade Automotiva, órgão responsável por sua tradução e comercialização no país.

Figura 1 - Fases do APQP Fonte: APQP (2016)

Existe vasta literatura sobre desenvolvimento de produto e sobre o processo de desenvolvimento colaborativo. No entanto, a revisão bibliográfica realizada no presente estudo encontrou pouca literatura publicada sobre a metodologia de trabalho aplicada nas montadoras de veículos americanas, o APQP (Planejamento Avançado da Qualidade do Produto).

Não foram encontradas pelos autores dessa pesquisa, publicações que se assemelhassem à proposta de estudo e discussão de metodologia do APQP estabelecida no corrente trabalho.

No entanto, o APQP, bem como metodologias similares utilizadas em outras montadoras, é um instrumento que guia a gestão de desenvolvimento de inúmeras e relevantes empresas, observando-se também as consideráveis semelhanças nas fases de desenvolvimento propostas por Clark e Fujimoto (1991).

2.1 *Softwar*e, definição e história

Softwares são instruções cuidadosamente organizadas através de códigos específicos, escritas por profissionais especializados, que permitem que equipamentos destinados ao processamento dessas informações, normalmente os computadores, estejam aptos a serem utilizados.

Basicamente, pode-se categorizar o *software* em duas esferas: a primeira, denominada sistema operacional (*Windows, Linux, IOS*, entre outros), que é responsável pelo controle, organização e administração de todos os recursos de um equipamento, permitindo a interação entre os usuários. A segunda se refere aos tipos de software, conhecidos como *softwares* de aplicação, que são programas específicos e especializados para cada tipo de atividade, como processamento de texto (*Microsoft Word*), imagens (*Adobe Photoshop*), tabelas (*Microsoft Excel*), entre outros.

Na década de 70, caracterizada pelo início da era digital e invenção dos microprocessadores, para desenvolver um *software* era necessário utilizar-se de uma técnica conhecida como programação estruturada. Em outras palavras, um método estrutural dividido em três partes: sequência, decisão e repetição, onde os desenvolvedores elaboravam apenas sub-rotinas e funções, de forma linear, com a lógica de desenvolvimento aplicada de cima para baixo, também conhecida por JSD, *Jackson Structures Programming*, em alusão ao seu criador Michael A. Jackson. Apesar de ainda ser usada para desenvolvimentos simples, diretos e rápidos, ela foi substituída na década de 90 pela *Object-oriented programming* (OOP), programação orientada a objetos.

Recentemente, a partir dos anos 2000, foram desenvolvidos diversos modelos de desenvolvimento de software com características distintas, porém todos em conformidade com o novo modelo de desenvolvimento de software, o qual é chamado de AUP, *Agile Unified Process*, ou seja, Processo Ágil Unificado.

Serão usadas as fases padrões de desenvolvimento de software como modelo de eficiência, onde será adotado como base o modelo em “cascata”, proposto por Royce (1970), que possuem um padrão sequencial de desenvolvimento, onde todos os passos da etapa anterior são concluídos antes que a próxima se inicie. Vale lembrar que o modelo atual leva em conta o *feedback* de cada fase, influenciando as próximas para um melhor resultado (figura 2).

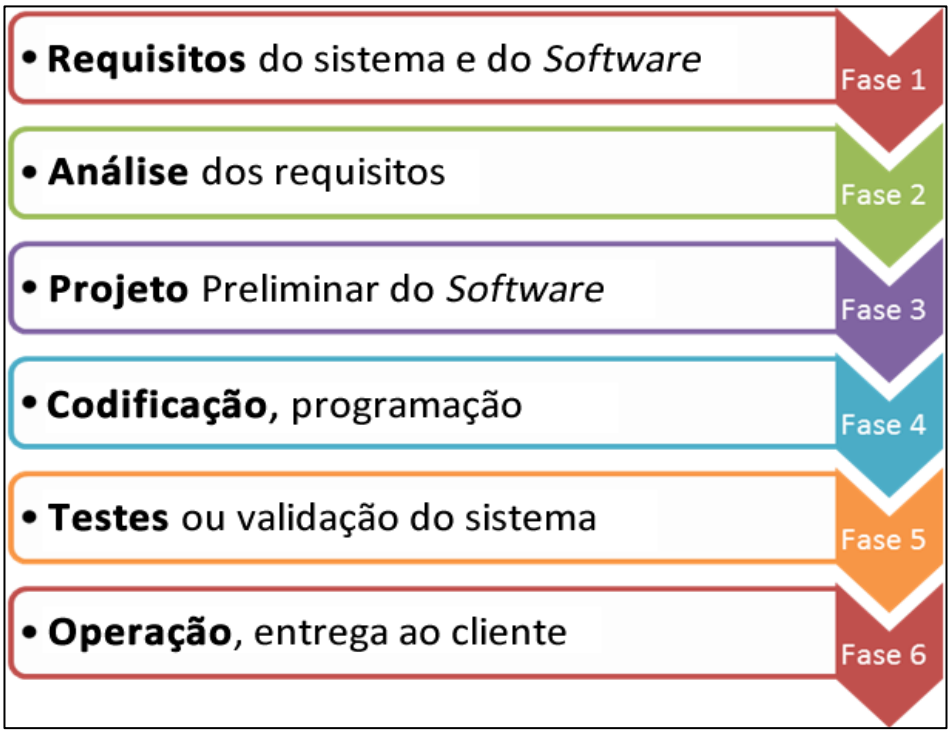

Figura 2 - Modelo em "cascata" ou Waterfall model Fonte: Royce (1970)

Para elucidar todos os pontos do embasamento teórico dado nessa pesquisa, classificam-se como desenvolvedores os profissionais que estão envolvidos no desenvolvimento de uma aplicação, um *software* para computador.

2.2 Qualidade de *Software* segundo a ISO

Assim como em grande parte dos processos produtivos, a utilização de modelos e padrões internacionais de desenvolvimento que visem agregar melhorias substanciais ao resultado final, pode trazer resultados significativos no processo de desenvolvimento de *softwares*, influenciando diretamente na qualidade final do produto a ser entregue ao cliente, bem como no aumento da produtividade dos profissionais envolvidos durante o desenvolvimento do mesmo, gerando uma relação inversamente proporcional entre custo e valor agregado, de maneira que o custo seja sempre inferior ao valor do produto final.

No entanto, conforme exposto no item 2.2, existem vários padrões de desenvolvimento de *software*, sendo assim, para garantir que empresas desenvolvedoras de *software* tenham um processo de desenvolvimento adequado e que garanta certo grau de confiança e credibilidade aos compradores, foram criadas certificações como, por exemplo, a MPS.BR, que avaliam se o produto final atende às normas técnicas. Quando a avaliação tem resultado positivo, a empresa desenvolvedora recebe, por meio de uma certificação, o aval para elaboração do projeto, agregando valor e qualidade em seu produto final, o *software*.

Nessa pesquisa foi considerada como referência a norma técnica ABNT NBR ISO/IEC 12207:2009 - Engenharia de sistemas e *software* – Processos de ciclo de vida de *software*, que estabelece uma estrutura comum para os processos e atividades de desenvolvimento de *software* e a da norma ABNT NBR ISO/IEC 25010:2011 - *Systems and software engineering - Systems and software Quality Requirements and Evaluation* (SQuaRE) - *System and software Quality models.*

2.3 ABNT NBR ISO/IEC 12207:2009 Engenharia de sistemas e *software*

A ISO/IEC 12207, desenvolvida pela *International Organization of Standardization*, em português, Organização Internacional de Normalização (ISO), é o documento internacional que exprime através de seu conteúdo as especificações técnicas, critérios, diretrizes e definições, utilizadas no processo de desenvolvimento de *software.*

Esta norma agrupa as atividades que podem ser executadas durante o ciclo de vida de *software*, fornecendo um conjunto abrangente de processos agrupados em três amplas classes: as fundamentais, as de apoio e as organizacionais, sendo cinco processos na classe fundamental, oito processos de apoio e quatro processos organizacionais, que visam ajudar empresas a compreenderem todos os componentes presentes na aquisição e fornecimento de *software* e, assim, estarem aptos a firmar contratos e executarem projetos de forma mais eficaz (figura 3).

Figura 3 - Processos do ciclo de vida Fonte: ABNT NBR ISO/IEC 12207:2009

2.4 Processos fundamentais de ciclo de vida.

- Processo de aquisição; Processo de fornecimento; Processo de desenvolvimento; Processo de operação; Processo de manutenção.

2.5 Processos de apoio de ciclo de vida

- Processo de documentação; Processo de gerência de configuração; Processo de garantia da qualidade; Processo de verificação; Processo de validação; Processo de revisão conjunta; Processo de auditoria; Processo de resolução de problema.

2.6 Processos organizacionais de ciclo de vida

- Processo de gerência; Processo de infraestrutura; Processo de melhoria; Processo de treinamento.

2.7 ABNT NBR ISO/IEC 25010:2011 Engenharia de *software*

É uma norma da qualidade de produto de *software*, composta por oito características, duas a mais do que a norma anterior (NBR ISO/IEC 9126:2003**),** que se relacionam entre si proporcionando uma terminologia consistente para a análise da qualidade do *software* (tabela 1).

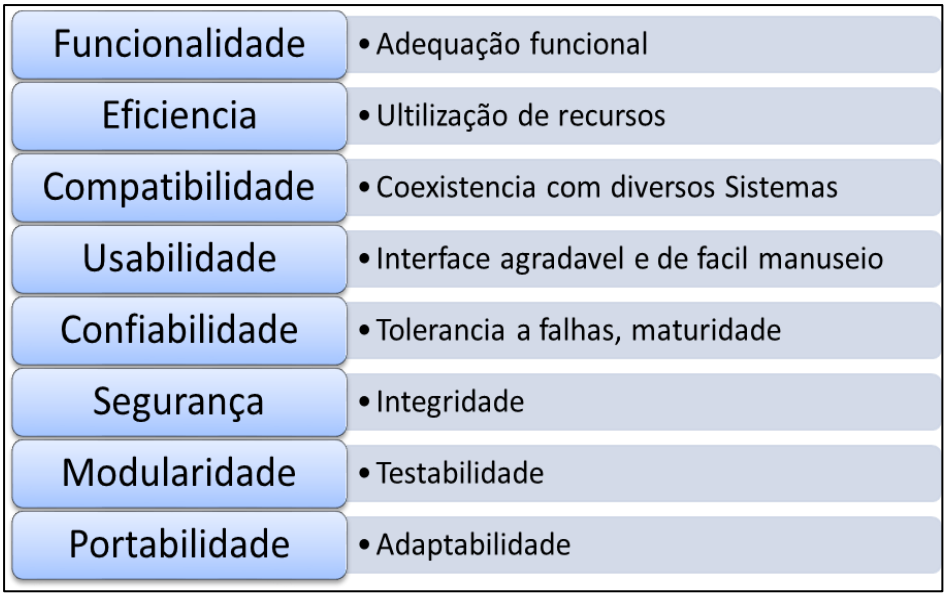

Tabela 1 - Oito características da ISO 25010 Fonte: ABNT NBR ISO/IEC 25010:2011

2.8 Segurança da Informação

Quando se fala sobre segurança da informação, também conhecida como InfoSec, se está falando em proteger as informações, ativos de uma empresa, no que diz respeito a sua confidencialidade, integridade e disponibilidade.

Quando se fala em confidencialidade, se está protegendo as informações no que se refere a sua divulgação para terceiros não autorizados. Já para a integridade das informações, se está protegendo essas informações para que não sejam alteradas por terceiros não autorizados e sua disponibilidade refere-se à disponibilidade dessas informações para pessoas autorizadas somente quando solicitadas.

2.9 Plano de Contingência

Em Tecnologia da Informação, um plano de contingência, também conhecido como *Disaster Recovery Plan* (DRP), é um plano reativo com diretrizes que devem ser tomadas quando operações são interrompidas por qualquer tipo ação, seja interna ou externa, assegurando a continuidade dos negócios.

Um bom plano de contingência relacionado com software leva em conta o *backup*, uma cópia representativa de todo código e banco de dados em um momento específico e sua recuperação, o *recovery*.

Uma boa estratégia de *backup* é essencial para a continuidade dos negocios, além de servir como proteção de dados em caso de falha de hardware, exclusões acidentais ou desastre, ou mesmo proteção contra alterações não autorizadas feitas por um intruso.

Um bom sistema de *Backup e Recovery* consiste em uma sequência de atividades interativas que necessitam de monitoramento e controle, descritas a seguir:

- Planejar e Preparar; Identificar os requisitos do backup; Selecionar e desenvolver uma estratégia de backup; Implementar e aplicar essa estratégia de backup; Monitorar a estratégia; Testar recorrentemente a recuperação de dados.

Um bom sistema de backup, implementado, confiável, testado e redundante, permitirá o sucesso do plano de contingência e por consequência, um aumento na possibilidade de continuidade dos negócios.

2.10 Planejamento de Continuidade do Negócio

O planejamento de continuidade de negócio, também conhecido como BCP - *Business Continuity Planning*, envolve o desenvolvimento de um plano de contingência baseado em um relatório de análise de impactos nos negócios, conhecido como BIA - *Business Impact Analisys.*

No caso de interrupção dos processos organizacionais de uma empresa, ele foi projetado para garantir a recuperação das atividades empresariais em caso de falhas críticas em um nível aceitável de operação dentro de um prazo pré-definido minimizando, assim, o impacto das perdas para a organização.

O planejamento de continuidade dos negócios é um processo cíclico que envolve os seguintes passos: Identificação crítica das atividades da empresa; Avaliação de risco de continuidade dos negócios; Desenvolvimento de um plano de Continuidade; Plano de aprovação e implementação.

2.11 Quinto poder

O Quinto poder é um termo controverso cuja pretensão inicial foi desenvolver um sistema de separação de poderes do governo proposto por Montesquieu em 1748 (QUINTO PODER, 2016). Existindo tal poder de caráter não oficial, há alguns candidatos a este quinto poder: um deles, objeto deste estudo, é a *Internet*.

A *internet* é um poderoso gerador de debates por vezes não democráticos, mas que atua na comunicação de massa e que promove a globalização. Não tem governança centralizada ou políticas de acesso e uso, sendo que cada rede constituinte define suas próprias políticas, impactando diretamente na educação, governos, publicidade e na ética, onde se verificam os crimes pela *internet* (*cyber-crime*) cometidos pelos *hackers* (HACKER, 2016).

Em informática, *hacker* é um indivíduo que se dedica intensamente em conhecer e modificar os aspectos mais internos de dispositivos, programas e redes de computadores.

Graças a esses conhecimentos, um *hacker* frequentemente consegue obter soluções e efeitos extraordinários, que extrapolam os limites do funcionamento "normal" dos sistemas como previstos pelos seus criadores, podendo alterá-los, mudar suas funções, acessar informações, efetuar espionagem industrial ou, até mesmo, remover o sistema de operação.

Na *Deep Web*, uma *internet* de quarto nível não indexada e oculta, utilizada normalmente por pessoas que tem a necessidade de manter o sigilo de informações e o anonimato, é onde são encontrados *hackers* dispostos a compartilhar informações para derrubar sites, expor dados, cometer fraudes, sequestros de dados, chantagem, falsificação ou mesmo apropriação indevida, também conhecidos como *Cyber-crimes*, utilizando para isso a *internet* como ferramenta de base de ataque.

Vale lembrar que *hackers* podem ser profissionais de segurança da informação que atuam para melhorar a segurança de sistemas computacionais ou *cyber* criminosos: tudo vai depender do caráter do mesmo.

2.12 *Penetration Test - Pentest*

O *Pentest*, também conhecido como teste de Intrusão, normalmente executado por um profissional de segurança da informação, tem como objetivo encontrar vulnerabilidades em sistemas, *software*, redes e empresas, aumentando a segurança desses sistemas (PENTEST, 2016).

Esses testes são executados sem dados de acesso, logins, senhas ou qualquer tipo de informação que facilite este trabalho, pois a ideia é justamente que se descubram vulnerabilidades expostas pelo sistema em estudo.

A principal diferença entre um *Pentester* (testador de penetração) e um *Cyber* criminoso é a permissão. O Pentester terá a permissão do proprietário para acessar os recursos de computação que estão sendo testados e será responsável para fornecer um relatório com as vulnerabilidades encontradas e as possíveis soluções para resolver cada uma delas.

3. Metodologia

A metodologia é fundamentada na associação entre os fatores definidos com base na eficiência, eficácia, satisfação do cliente e segurança, que impactarão diretamente no ciclo de vida do *software*, compreendendo os processos desde sua concepção, passando por sua manutenção e reconquista de clientes, segurança da informação, plano de contingência, até a continuidade dos negócios em caso de ataques cibernéticos, *cyber crimes*, e que, através de índices matemáticos que possam quantificar tais fatores, pode-se estabelecer uma base genérica para análise e melhoria contínua do ciclo de vida de um *software* e, também por intermédio deste indicador numérico, determinar a propensão de sucesso do mesmo.

Esta metodologia baseia-se em pontuações adquiridas por meio de indicadores matemáticos chave, chamadas *Key Performance Indicators* – KPI. Estes, por sua vez, são baseados nos tópicos enumerados na tabela 2, de forma a prover uma conversão de características qualitativas do processo, resultando em indicações quantitativas que serão analisadas por meio de uma matriz matemática pré-definida de resultados.

3.1. Aplicação da Metodologia

Levando em conta que eficiência está diretamente relacionada à gestão dos recursos e a eficácia tem a mesma relação com resultados, esta nova metodologia está estruturada em pilares básicos, nomeados como 2E2S. Dentro de cada um destes pilares, foram categorizados os princípios apresentados no quadro teórico e, baseadas nesta categorização, foram definidas pontuações específicas que servirão como métricas matemáticas universais do processo durante a realização do mesmo.

Como exposto anteriormente, os pilares da metodologia proposta são:

- Eficiência baseada na gestão de recursos;
- Eficácia baseada em resultados;
- Satisfação do Cliente levando em conta suas avaliações e Feedbacks;
- Segurança (2E2S), baseado na credibilidade da Tecnologia da Informação empregada no projeto.

3.2. Pilares do 2E2S

Com o intuito de esclarecer cada métrica exercida no decorrer do processo de avaliação, serão enumerados os relacionamentos estabelecidos para formulação de cada KPI, bem como a justificativa para o grau de importância de cada indicador no resultado final.

- Eficiência: São considerados como elementos substanciais na composição do indicador de eficiência todos os fatores que tem relação, mesmo que mínima, com a gestão de recursos utilizados no projeto. Assim, considerando as fases de desenvolvimento do APQP, foram elencadas as etapas de Planejamento e Projeto, que englobam a gerência de Requisitos, Análise, Projeto e Codificação (fases do macroprocesso de desenvolvimento do Software), além das características de Funcionalidade, Eficiência, Compatibilidade, Usabilidade e Modularidades, presentes na ISO 25010 e a combinação com os subprocessos de Aquisição, Fornecimento, Documentação e Definição de Infraestrutura. O agrupamento de todos estes elementos definirá os quesitos necessários para pontuação da Eficiência;

- Eficácia: Da mesma maneira, foram organizadas todas as hipóteses relacionadas a resultados dentro deste elemento, iniciando pelas etapas de Verificação e Validação do APQP, complementando-as com as fases de Testes e Operações do desenvolvimento geral de *software*, que também foram equalizadas à mesma categoria os processos de Operação, Verificação e Validação presentes na ISO 12207;

- Satisfação do Cliente: Para definição do índice relacionado a este fator, levaram-se em conta todas as atividades e processos capazes de recolher opiniões, experiências e Feedbacks dos clientes, claramente expostos nas fases de *Feedback* e Ações Corretivas do APQP, confiabilidade da ISO 25010 e processos "pós" entrega da ISO 12207, como Manutenção, Revisão, Auditoria, Resolução de Problemas, Melhorias e Treinamentos;

- Segurança: Com o avanço iminente do que foi definido anteriormente como quinto poder, tornaram-se fatores primordiais os sistemas de segurança utilizados no decorrer de todo o ciclo de desenvolvimento e vida do *software*. Assim, pode-se destacar este fator como inerente a todos os outros pilares levantados nos itens anteriores, ocorrendo de maneira paralela e concomitante, porém influindo de maneira muito decisiva para o alcance de bons índices ao final do processo de análise, além da continuidade dos negócios.

O *softwar*e deve estar preparado com um plano de contingência para ser utilizado caso tenha alguma situação de interrupção, como um problema no *hardware*, um ataque cibernético, entre outros.

Para garantir a sobrevivência do mesmo é necessária a gestão de Continuidade de Negócios (GCN), que é uma parte da gestão de riscos estratégicos e uma necessidade básica da gestão moderna, que serve para proteger investimentos, marcas, pessoas, tecnologias e informações, aumentando a resiliência empresarial.

Não se trata apenas de garantir a continuidade da tecnologia da informação e comunicação (TIC) que suporta o negócio: se trata de agregar valor à Governança Corporativa e fornecer meios e informações para a proteção da viabilidade do negócio

3.3. Considerações Adicionais sobre a Metodologia

Conforme o modelo de cascata proposto por Royce (1970), serão alinhadas as etapas do processo APQP buscando a redução dos possíveis modos de falha e também a minimização dos riscos para o 2E2S.

Para isso serão trabalhadas as principais fases do APQP, que são o Planejamento, Projeto, Verificação, Validação, *Feedback* e Ações Corretivas pois, segundo Rozenfeld (2006), o processo de desenvolvimento do produto, em particular suas primeiras fases, é fundamental para determinar todo o custo do projeto, inclusive o custo do produto final: neste caso, o desenvolvimento de um *software.*

Sendo assim, compreendidas as necessidades do cliente, que neste caso é um contratante adquirindo um software para a gestão de sua organização, serão aplicadas as cinco fases do APQP relacionadas a seguir:

- Executar o Planejamento, que seria a escolha das metodologias utilizadas;
- Verificar e organizar os dados colhidos em campo e fornecidos pelo contratante;
- Validar todos os processos por meio de testes no sistema e na segurança;

- Usar como Feedback os resultados obtidos na implementação do sistema junto aos usuários;
- Desenvolver as Ações Corretivas, que serão possíveis para realizar correções e adequações no software, além da melhoria em todo o ciclo de desenvolvimento do software.

FASES APQP	FASES *SOFTWARE*	ISO 25010	ISO 12207
Planejamento	Requisitos	Funcionalidade	Aquisição
		Eficiência	Fornecimento
	Análise	Compatibilidade	Documentação
		Usabilidade	Infraestrutura
		Modularidade	
Projeto	Projeto	**SEGURANÇA**	Desenvolvimento
			Configuração
	Codificação		Qualidade
			Gerência
Verificação	Testes		Operação
Validação	Operação		Verificação
			Validação
Feedback		Confiabilidade	Manutenção
			Revisão
			Auditoria
Ações Corretivas			Resolução de Problema
			Melhoria
			Treinamento

Tabela 1 – Matriz Matemática para Cálculo 2E2S

3.4. Indicadores matemáticos de desempenho

Esta metodologia está estruturada em pilares básicos, 2E2S. Em cada um destes pilares se encontram alguns dos princípios de cada fundamento definindo pontuações específicas, que servirão como indicadores matemáticos para a aplicação da metodologia.

- Os indicadores matemáticos de Eficiência (E1), que são baseados na gestão de recursos, são: Planejamento, desenvolvimento, funcionalidade e documentação;
- Os indicadores matemáticos da Eficácia (E2), que são baseados em resultados, são: Verificação e validação;
- Os indicadores matemáticos de Satisfação (S1) do Cliente são: Feedbacks, ações corretivas, confiabilidade e melhorias;
- Os indicadores matemáticos de Segurança (S2), baseados na credibilidade da Tecnologia da Informação empregada no projeto, são: Confidencialidade, Integridade, Disponibilidade, Vulnerabilidade, Plano de Contingência, Planejamento de Continuidade do Negócio e Pentest.

3.5. Validação da Metodologia

Com o propósito de verificar a aplicabilidade da metodologia 2E2S, optou-se por realizar uma auditoria dos processos do *software QualityManager®*, em prol da identificação dos pontos falhos (e de melhoria) que pudessem contribuir para definição das estratégias de crescimento da ferramenta.

Através dos pilares definidos na metodologia, será utilizada a pontuação obtida no final da análise crítica por meio da matriz analítica para posicionar o *software QualityManager®* de acordo com sua performance ao longo do processo, utilizando-se de todos os conceitos enumerados para avaliação dos pontos falhos e de melhoria dentro da ferramenta utilizada no estudo de caso. Assim, pretende-se identificar especificamente os pontos responsáveis pelo comprometimento do projeto.

3.6. Aplicação do 2E2S no *Software QualityManager®*

Como parte da metodologia de análise, para realizar a conversão dos índices qualitativos em quantitativos, é necessário avaliar cada indicador matemático de maneira binária, atribuindo nota 1 para os indicadores matemáticos que foram considerados satisfatórios e 0 para os que não atendem o requisito.

Para cada parâmetro, precisam ser somados os resultados encontrados e, em seguida, fazer a divisão do resultado pelo número de indicadores matemáticos utilizados.

Cada pilar corresponde à somatória de todos os seus indicadores matemáticos internos, totalizando no máximo 25% do processo total.

1º passo – Cálculo matemático dos indicadores

$$(E1.1 + E1.2 + E1.3 + E1.4) / 4 = E1 \quad (1)$$

$$(E2.1 + E2.2) / 2 = E2 \quad (2)$$

$$(S1.1 + S1.2 + S1.3 + S1.4) / 4 = S1 \quad (3)$$

$$(S2.1 + S2.2 + S2.3 + S2.4 + S2.5 + S2.6) / 6 = S2 \quad (4)$$

2º passo – Cálculo matemático do índice 2E2S

Somar as equações: eq. (1) + eq. (2) + eq. (3) + eq. (4) e dividir por 4.

$$(E1 + E2 + S1 + S2) / 4 = X \quad (5)$$

Multiplicar o resultado da eq. (5) por 100 obtendo assim o índice 2E2S.

$$X * 100 = \text{Índice 2E2S} \quad (6)$$

Com base no resultado da eq. (6), verificar na tabela 2 em qual intervalo o índice se enquadra e a sua classificação segundo a metodologia proposta.

Nível	Intervalo	Descrição
1	0 a 25%	Mude de ramo ou profissão
2	26 a 50%	Você precisa contratar profissionais qualificados
3	51 a 75%	Está faltando um pouco de dedicação
4	76 a 86%	A vida útil de seu *software* deve acabar em breve
5	87 a 95%	Muito bem, você vai chegar lá!
6	96 a 99%	Parabéns, você tem um ótimo produto.
7	100%	Seu produto atingiu a EXCELÊNCIA.

Tabela 2 - Classificação 2E2S

A análise de aplicação do método consiste na validação precisa dos pontos críticos que compõem a elaboração dos resultados aplicados no gráfico analítico 2E2S, diagnosticando a propensão de sucesso do *software*. Desta forma, os resultados deste estudo serão elencados por tópicos, dispostos de maneira clara e objetiva, visando a contribuição nas decisões estratégicas da empresa detentora/desenvolvedora.

Seguindo a cronologia da metodologia, será apresentada uma árvore dos pontos notórios do processo, desde sua fundamentação tecnológica até a entrega e satisfação ao cliente final.

3.7. Resultado da metodologia no *QualityManager*®

Após concluir a análise dos processos do *software QualityManager*®, foi elaborado o relatório da tabela 3.

PILAR 2E2S	INDICADOR	NOTA	MÉDIA
Eficiência (E1)	Planejamento	1	0,75
	Desenvolvimento	1	
	Funcionalidade	1	
	Documentação	0	
Eficácia (E2)	Verificação	1	1
	Validação	1	
Satisfação (S1)	*Feedbacks*	0	0,75
	Ações corretivas	1	
	Confiabilidade	1	
	Melhorias	1	
Segurança (S2)	Confidencialidade	1	1
	Integridade	1	
	Disponibilidade	1	
	Vulnerabilidade	1	
	Plano de Contingência	1	
	Planejamento de Continuidade do Negócio	1	
		Média Final	3,5/4
		ÍNDICE 2E2S	87,5 %

Tabela 3. Resultado 2E2S *QualityManager*®

4. Considerações Finais

A qualidade de um *software* não se atinge de forma natural. Além de um bom processo para desenvolvê-lo baseado nas melhores práticas da ISO, sua implementação e sustentabilidade (*disaster recover*) são partes importantes para a longevidade desta aplicação.

Quando se fala sobre segurança da informação, isso significa proteger as informações no que diz respeito à sua confidencialidade, integridade e disponibilidade. Não se trata apenas de garantir a continuidade da tecnologia da informação e comunicação (TIC) que suporta o negócio, mas se trata de agregar valor à Governança Corporativa e fornecer meios e informações para a proteção da viabilidade do negócio.

Como demonstrado, a metodologia 2E2S auxilia matematicamente na organização das informações dos metodos de desenvolvimento listando e registrando, de forma organizada e abrangente, as fases de desenvolvimento de software. Acrescenta ainda um terceiro fator que é a segurança, relacionada diretamente com a sustentabilidade do negócio.

A utilização desta metodologia auxilia matematicamente na identificação do que pode ser considerado o fator determinante da ocorrência do sucesso, fator este que por vezes pode ser ocultado pela falta de atenção em todos os aspectos que ocorrem dentro do ciclo de vida do software.

Verificou-se ainda que, após a aplicação do método 2E2S no *software QualityManage*r®, foram identificados seus principais problemas e falhas, além de se perceber a necessidade de criar planos de contingência mais eficazes e aumentar o fator segurança com relação à programação e a segurança das informações.

Como visto, é necessária a implantação da metodologia 2E2S de forma cultural na empresa, realizando-a periodicamente, com o propósito de detectar novos pontos falhos antes que possam causar quaisquer prejuízos à estrutura geral do *software*. Desta forma, após a correção das falhas apontadas no relatório da autoria do *software*, foi realizada uma nova auditoria baseada na metodologia 2E2S, verificando se o *QualityManager*® atingiu a excelência segundo o 2E2S, ou se serão necessários outros planos de ações para que se possa considerar que o software atingiu o ciclo de vida sustentável.

Através do exposto nessa pesquisa, um *software* obteve sua excelência quando todo seu ciclo de desenvolvimento foi analisado sob a ótica da eficácia, da eficiência, da satisfação do cliente e da segurança ao mesmo tempo, pois são elas que dão a sustentabilidade para os negócios da empresa. Mais do que isso, o aperfeiçoamento das técnicas para que os índices sejam mantidos próximos aos índices máximos, contribui extraordinariamente para a continuidade dos negócios, uma vez que a vida cíclica do *software* será sempre preservada.

Para a utilização do 2E2S, deverá ser contemplado o segmento de mercado no qual a organização está inserida, merecendo maior atenção em casos especiais, entendidos como atípicos, onde existem regras intrínsecas para cada segmento de mercado.

Vale lembrar que o processo de desenvolvimento de *software* é, antes de tudo econômico, e o profissional que irá aplicar a metodologia deverá se valer da criatividade, sempre com bom senso e equilíbrio.

A metodologia proposta neste trabalho tem uma abordagem experimental para tentar explicar um raciocínio dedutivo, onde a conclusão ratifica apenas as premissas iniciais requerendo um longo e contínuo trabalho para se chegar a um bom resultado.

Para fins de melhoria e agregação de valor ao 2E2S, foi criado o endereço para compartilhamento desta metodologia – http://www.2e2smodel.com/ – e assim, a partir do conhecimento da comunidade, serão acrescidas novas ideias e possibilidades que consequentemente resultarão em um aumento da confiabilidade e abrangência deste método analítico. Com esta medida, é possível conservar e corroborar, cada vez mais veementemente, a premissa de melhoria contínua da qualidade processual.

Ainda que, num primeiro momento, o foco esteja no desenvolvimento de aplicações,

existe a possibilidade de estender estes parâmetros de análise a outras áreas de desenvolvimento, utilizando como argumento o caminho lógico adotado para elaboração deste estudo.

Referências

APQP. *Advanced Product Quality Planning ou Planejamento Avançado da Qualidade do Produto.* Origem: Wikipédia, a enciclopédia livre. Disponível em: http://pt.wikipedia.org/wiki/Advance_Product_Quality_Planning. Acesso em: 12 Março 2016.

ASSOCIAÇÃO BRASILEIRA DE NORMAS TÉCNICAS. NBR ISO/IEC 12207. *Engenharia de sistemas e software - Processos de ciclo de vida de software.* Válida a partir de Abril de 2009.

AUTOMOTIVE INDUSTRY ACTION GROUP (AIAG). *Advanced Product Quality Planning and Control Plan*, 2ª edição, 2008. EUA.

CHIAVENATO, I. *Recursos humanos na Empresa: pessoas, organizações e sistemas.* 3.ed. São Paulo: Atlas, 1994. p. 67-76.

CLARK & FUJIMOTO, T. *Product development performance: strategy, organization and management in the world auto industry*. Boston: Harvard Business School Press, 1991.

DRUCKER, Peter F. *Introdução a Administração*. 2ª reimpressão da 1° ed. de 1.988. São Paulo: Thomson, 2002.

HACKER. Disponível em https://pt.wikipedia.org/wiki/Hacker. Acessado em 10 de Junho de 2016.

IQA - INSTITUTO DA QUALIDADE AUTOMOTIVA. *Planejamento Avançado da Qualidade do Produto (APQP) e Plano de Controle Manual de referência*. 2 ed. Michigan: AIAG, 2008.

INTERNATIONAL STANDARD ORGANIZATION. **NBR ISO/IEC 25010**. *Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE)* - System and software quality models. Válida a partir de Março de 2011.

ISO/IEC 12207. Origem: Wikipédia, a enciclopédia livre. Disponível em: https://pt.wikipedia.org/wiki/ISO/IEC_12207. Acesso em: 13 Abril 2016.

ISO/IEC 25010:2011 *Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) - System and software quality models.* Origem: ISO Store. Disponivel em: http://www.iso.org/iso/catalogue_detail.htm?csnumber=35733. Acesso em: 1 Maio 2016.

PENTEST, Penetration Testing: *Assessing Your Overall Security Before Attackers Do*. SANS Institute InfoSec. June 2016.

QUALITYMANAGER®, *Software. Software para gestão da qualidade.* Disponível em http://www.qualitymanager.com.br, 2016.

QUINTO PODER. Disponível em https://pt.wikipedia.org/wiki/Quinto_poder. Acessado em 10 de Junho de 2016.

ROYCE, Winston W. *Managing the Development of Large Software Systems, Proceedings of IEEE WESCON* 26 (August): 1–9, 1970.

ROZENFELD, H.; FORCELLINI, F. A.; AMARAL, D. C.; TOLEDO, J. C. e outros – *Gestão de desenvolvimento de produtos: uma referência para melhoria do processo.* São Paulo: Editora Saraiva, 2006.