Um Levantamento de Requisitos de Processos para Certificação de Sistemas Militares com Software

Johnny Marques[1], Sarasuaty Yelisetty[1], Cecília Dutra[2], Fernando Sulzbacher[2]
[1]Instituto Tecnológico de Aeronáutica (ITA), São José dos Campos/SP – Brasil
[2]Instituto de Fomento e Coordenação Industrial (IFI), São José dos Campos/SP – Brasil

***Resumo* – O desenvolvimento de softwares embarcados militares necessita certificação e deve cumprir com exigências de segurança e missão. Nos últimos anos, estes tipos de sistemas tornaram-se mais complexos, o que resultou na adoção de padrões e processos de software mais rigorosos. Especialmente na aviação militar, existe uma tendência de utilização de padrões e processos aplicados com sucesso na aviação civil, como a norma RTCA DO-178C. Este artigo apresenta um levantamento dos requisitos para a certificação de Sistemas Militares com Software (SMS), visando a garantia de qualidade do produto de software com segurança e cumprimento da missão atribuída.**

***Palavras-Chave* – Software, certificação, militar, sistemas, requisitos.**

I. INTRODUÇÃO

A aviação civil tem como guia principal a segurança de voo, assim, todo o desenvolvimento de software não visa só um aumento de funcionalidades. Principalmente nos grandes jatos, o uso de tecnologias avançadas incorporadas aos sistemas traduz bem a necessidade de softwares que atendam a um alto grau de tecnologia, combinado com a preocupação em segurança [1].

A aviação de defesa possui um outro foco sobre o software embarcado concentrando primariamente no cumprimento das missões, nas inovações tecnológicas (novas ferramentas, processos e tecnologias) e mantendo segurança.

De acordo com o entendimento do Comando da Aeronáutica, e previsto na ICA -57-21 [2], o conjunto de requisitos, proposto pelo contratada e aceito pelo certificador, deve traduzir o entendimento comum de quais características o produto deve possuir de modo a garantir a segurança e o cumprimento da missão. No mercado de defesa, o desenvolvimento de software pode ser definido por três grandes características [1]: missão; novas tecnologias; e conteúdo preservado, já que envolve informações sensíveis de forças militares de diversos países, na maior parte das vezes.

As aeronaves de defesa possuem um conjunto de características bem definidas, diferente do transporte normal de passageiros. Uma aeronave de defesa, quando projetada, normalmente atende a uma força militar de algum país. Assim, normalmente é definido um conjunto de normas entre fabricante e força militar, para definir o sigilo de informações de determinados sistemas embarcados.

J. Marques, johnny@ita.br; S. Yelisetty, sara.mhy@gmail.com; C. Dutra, ceciliacmdm@ifi.cta.br; F. Sulzbacher, sulzbacherfs@ifi.cta.br

Por outro lado, as abordagens tradicionais de software e engenharia de sistemas são normalmente realizadas em projetos que desenvolvem soluções personalizadas individualmente e atingem seus limites quando confrontadas com aumento de complexidade e variabilidade, em um ambiente com clientes cada vez mais exigentes com redução de custos e aumentos de qualidade e produtividade [3].

O objetivo principal deste trabalho é apresentar um levantamento de Requisitos de Processos que podem ser utilizados na certificação de Sistemas Militares com Software (SMS) com uso intensivo de software, evitando assim a necessidade de prescrever uma norma ou modelo de maturidade específico na ocasião da contratação do desenvolvimento do sistema.

Além desta seção 1, o trabalho inclui mais outras 7 seções. A seção 2 descreve, suscintamente as principais normas e modelos de maturidade aplicáveis. A seção 3 apresenta os trabalhos correlatos. A seção 4 sintetiza o processo de certificação de SMS. A seção 5 apresenta a metodologia de pesquisa aplicada. A seção 6 apresenta os principais processos de software. A seção 7 apresenta a principal contribuição deste trabalho, com os 22 requisitos de processos propostos pelos autores. A seção 8 apresenta a conclusão e trabalhos futuros dos autores.

II. NORMAS E MODELOS DE MATURIDADE DE SOFTWARE

Criada pela Radio Technical Commission for Aeronautics. (RTCA), o padrão DO-178C [4] estabelece os procedimentos necessários para que o fabricante de aeronaves com uso de software embarcado demonstre que seu produto foi construído atendendo aos requisitos de aeronavegabilidade [5]. A DO-178C foi desenvolvida para estabelecer considerações para desenvolvedores, instaladores, e usuários quando o projeto de um equipamento aeronáuticos é implementado usando software [6].

A ISO/IEC/IEEE 12207 [7] estabelece uma estrutura comum para processos de ciclo de vida de software, com terminologia bem definida, que pode ser referenciada pela indústria de software. A ISO/IEC/IEEE 12207 descreve 25 processos de Software: Implementação de Software, Análise de Requisitos, Projeto Arquitetônico, Projeto Detalhado, Construção, Integração, Teste de Qualificação, Gerenciamento de Documentação, Gerenciamento de Configuração, Garantia da Qualidade, Verificação, Validação, Revisão, Auditoria e Resolução de Problemas.

A IEC 61508 [8] consiste em um padrão de segurança funcional aplicável em qualquer domínio de Software em Ambientes Regulados (SAR). Dentre o conjunto de partes da IEC 61508, a terceira parte (IEC 61508-3) apresenta os detalhes sobre o ciclo de vida de Software, baseado no Ciclo de Vida em V. A IEC 61508-3 aborda o chamado Safety Integrity Level (SIL), onde este varia de 1 até 4 e para cada

um destes existe um conjunto de atividades associadas. A caracterização das atividades possui natureza de dois tipos: “altamente recomendadas” e “recomendadas”.

O *Capability Maturity Model Integration* (CMMI) é um modelo de referência que contém práticas (Genéricas ou Específicas) necessárias à maturidade em disciplinas específicas como: Engenharia de Sistemas, Engenharia de Software, Desenvolvimento Integrado de Processo e Produto, e Seleção de Fornecedores. O CMMI [9] foi baseado nas melhores práticas para desenvolvimento e manutenção de produtos.

O MPS-BR [10] ou Melhoria de Processos do Software Brasileiro, é um modelo de qualidade de processo criado em 2003 pela Softex (Associação para Promoção da Excelência do Software Brasileiro) para melhorar a capacidade de desenvolvimento de software nas empresas brasileiras. Para a definição do MPS-BR levou em consideração normas e modelos internacionalmente reconhecidos como CMMI (*Capability Maturity Model Integration*), e nas normas ISO/IEC/IEEE 12207 [7] e ISO/IEC 15504 [11] e na realidade do mercado brasileiro de software. A implementação do MPS-BR exige a aplicação de vários processos referentes ao produto de software.

III. TRABALHOS CORRELATOS

A área militar apresenta produtos com forte inovação tecnológica e com grande esforço em pesquisa e desenvolvimento. No entanto, muitas vezes as pesquisas não são divulgadas em publicações internacionais de acesso público, já que possuem conteúdo bastante preservado, o que limita a identificação de um grande número de trabalhos correlatos.

Chen & Unewisse [12] apresentam os desafios de engenharia de sistemas militares no contexto de System-of-Systems (SoS). Neste trabalho, os autores abordam as principais dificuldades que um SoS militar enfrenta, com destaque para a forte prescrição de normas e padrões.

Kim et al. [13] aplicaram o processo de engenharia de confiabilidade de software ao desenvolvimento de software na indústria de defesa da Coréia do Sul. O trabalho envolveu o levantamento de métricas para todas as fases de processo de desenvolvimento de software.

Çiflikli et al. [14] apresentaram uma abordagem de desenvolvimento de software orientada por modelo a ser usada em um sistema militar de comando e controle. O objetivo dessa abordagem é melhorar os princípios de design, como variabilidade, extensibilidade e flexibilidade.

Avaliação integrada de suporte a software militar foi estudada no artigo de Li et al. [15], com o conceito e classificação de software militar descrito, a relação entre suporte de software e suporte de software analisado.

Zhou et al. [16] apresentam o relacionamento entre a análise de requisitos e os testes para SMS. Com enfoque na necessidade de demonstração da cobertura via testes de todos os requisitos de software.

Benedicenti et al. [17] relatam a experiência da aplicação de métodos ágeis no setor de defesa, caracterizada principalmente por softwares integrados e de missão crítica.

Tsoucamoto et al. [18] uma abordagem para desenvolvimento de software inspirada em práticas ágeis, para fins de certificação e foi utilizada para desenvolver um Computador de Missão (CM), responsável pelo gerenciamento dedicado e independente de diversos subsistemas instalados em uma plataforma aviônica militar de uma aeronave cargueira de grande porte.

IV. PROCESSO DE CERTIFICAÇÃO DE SOFTWARE MILITAR

O processo de certificação possui três papéis principais definidos: o contratante, a empresa que detém os direitos sobre o produto a ser certificado, a contratada, a empresa que desenvolve o produto e o certificador, entidade que irá avaliar a aderência aos requisitos necessários para certificar o produto apresentado.

O processo de certificação pode ser evidenciado pela demonstração de cumprimento com requisitos e padrões previamente estabelecidos. Existem algumas maneiras de avaliação do certificador, destaca-se, no entanto, os seguintes métodos de avaliação tipicamente utilizados: testes, auditorias e verificação documental.

Existem diversos aspectos que precisam ser considerados no processo de certificação de sistemas críticos. Oshana & Kraeling [19] apontam os seguintes aspectos que são relegados frequentemente:

- Falha ao identificar requisitos de segurança e de missão;
- Requisitos para o sistema e seus componentes não identificados adequadamente;
- Escopo do projeto muda constantemente;
- Não estabelecer um contato único com o certificador;
- Submeter a documentação apenas no término do projeto; e
- Não estabelecer apropriadamente a sequência de tarefas de certificação.

V. METODOLOGIA DE PESQUISA

Devido às características relevantes da missão a ser satisfeita por um projeto de uma aeronave de uso militar, com foco específico para cumprimento de missões contratualmente acordadas, os autores deste artigo entendem que são necessárias as definições de características mínimas, que os processos de desenvolvimento de software dos produtos inclusos do projeto militar.

Existem diversas normas internacionais disponíveis que possuem orientações importantes para o desenvolvimento de Software para sistemas complexos e com características safety-critical, como: RTCA DO-178C e a ISO/IEC/IEEE 12207. Essas normas vêm sendo utilizadas como uma importante referência para o desenvolvimento de produtos complexos, como os utilizados em aeronaves militares.

Os autores deste trabalho entendem que os sistemas de missão de uma aeronave militar são de extrema importância e

por isso torna-se necessário a devida avaliação de como os desenvolvimentos desses produtos de Software são organizados e garantem uma qualidade inerente ao produto desenvolvido.

A metodologia aplicada neste trabalho envolve 5 passos, onde 3 já concluídos. Inicialmente, buscou-se a identificação dos Requisitos de Processos a partir das Normas e Modelos existentes (Passo 1) e dos trabalhos existentes na pesquisa bibliográfica (Passo 2). Posteriormente, foram desenvolvidos os 22 Requisitos de processos a serem apresentados na Seção 5. A Fig. 1 apresenta o fluxograma dos passos previstos. Os Passos 4 e 5 serão considerados como parte dos trabalhos futuros.

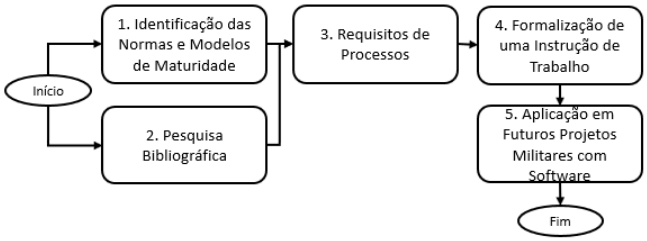

Fig. 1. Metodologia

VI. PROCESSOS DE DESENVOLVIMENTO DE SOFTWARE

O processo de requisitos captura e define requisitos de software do nível do produto que são implementados por software. Os requisitos de software normalmente identificam comportamento funcional e interfaces externas do software.

A necessidade de ter um processo de requisitos foi observada nos modelos CMMI e MPS-BR e nas normas DO-178C, IEC 61508-3 e ISO/IEC/IEEE 12207. A necessidade de um processo de requisitos robusto auxilia na eliminação de erros nos processos de arquitetura e serve para avaliar se a codificação foi feita corretamente, conforme observado no trabalho de Neiva et al. [20] e outros autores referenciados na Tabela 2.

O processo de arquitetura usa saídas do processo de Requisitos para desenvolver arquitetura de software, criando módulos e alocando funções expressas por requisitos de software. A necessidade de ter um processo de arquitetura foi observada nos modelos CMMI e MPS-BR e nas normas DO-178C e ISO/IEC/IEEE 12207. Assim, cada módulo é decomposto em componentes de software, definindo os dados e o fluxo de controle necessário entre eles, conforme observado nos trabalhos nos trabalhos de Stevanetic e Zdun [21] e outros autores identificados na Tabela 2.

O processo de projeto e codificação usa as saídas da fase de arquitetura para decompor cada módulo em componentes de software e para definir dados e controlar o fluxo necessário entre os componentes. A necessidade de um processo de projeto e codificação, assim como a necessidade do processo de implementação, foi observada nos modelos CMMI e MPS-BR e nas normas DO-178C, IEC 61508-3 e ISO/IEC/IEEE 12207. Conforme observado no trabalho de Contan et al. [22] e outros autores referenciados na Tabela 2, alguns requisitos de software de baixo nível que definem lógicas internas e detalhes que podem ser usados para implementar o código-fonte também são produzidos neste processo.

O processo de testes faz parte do desenvolvimento de software, e tem como principal objetivo revelar falhas/bugs para que sejam corrigidas até que o produto atinja a qualidade desejada e/ou acordada. A necessidade de um processo de testes foi observada nos modelos CMMI e MPS-BR e nas normas DO-178C, IEC 61508-3 e ISO/IEC/IEEE 12207. Conforme observado no trabalho de Dudila e Letia [23] e outros autores referenciados na Tabela 2, os testes são aplicados em três níveis: Unidade, Integração e Funcional.

O processo de validação é um processo para identificação de erros em todos os demais processos já tratados neste artigo. A necessidade de um processo de testes foi observada nos modelos CMMI e MPS-BR e nas normas IEC 61508-3 e ISO/IEC/IEEE 12207. Conforme observado no trabalho de Reddy e Prazad [24] e outros autores referenciados na Tabela 2, o processo de validação é iniciado com as revisões dos requisitos, passando pelas revisões da arquitetura, projeto, codificação e implementação (via inspeções do código) até chegar aos testes.

O processo de controle de configuração de software identifica e controla todos os artefatos gerados durante o desenvolvimento do software seguindo todos os processos já apresentados. A necessidade de um processo de controle de configuração foi observada nos modelos CMMI e MPS-BR e nas normas DO-178C, IEC 61508-3 e ISO/IEC/IEEE 12207. Conforme observado no trabalho de Wu e Hai Liu [25] e outros autores referenciados na Tabela 2, incluindo a captura de problemas e suas disposições.

O processo de planejamento e de fechamento da certificação são previstos na norma RTCA DO-178C e são necessidades obrigatórias no processo de certificação militar sumarizado na seção IV.

VII. REQUISITOS PARA PROCESSOS DE SOFTWARE MILITAR

A principal contribuição deste trabalho é o levantamento dos requisitos mínimos que devem ser utilizados em projetos militares com uso intensivo de software embarcado. Esses requisitos devem ser integrados e satisfeitos pelos processos definidos pela contratada do desenvolvimento de software para uso militar em seu processo de desenvolvimento de software. Os 22 requisitos de processos, são apresentados na Tabela 1.

A integração desses 22 Requisitos de Processos exige a definição de um ciclo de vida de software. Dentre os ciclos existentes na literatura, Pressman & Maxim [26], Sommervile [27] destacam que os principais ciclos existentes são:

— O Ciclo de Vida em Cascata (CVC);
— O Ciclo de Vida em V (CVV);
— O Ciclo de Vida Incremental e Iterativo (CVII); e
— O Ciclo de Vida em Espiral (CVE).

TABELA I. LISTA DOS 22 REQUISITOS DE PROCESSOS

Requisitos de Processo	*Descrição do Requisito*
RP 1	O seguinte conjunto de Processos de Software deve ser descrito num Plano de Desenvolvimento de Software (PDS) criado para demonstrar o Processo de Certificação, contendo: a) O processo de requisitos, incluindo como estes são revisados, rastreáveis para os requisitos de sistemas e com evidências e critérios que comprovam isso; b) O processo de arquitetura, incluindo sua descrição, os módulos, componentes, entradas e saídas de cada elemento da arquitetura; c) O processo de projeto e codificação, incluindo a definição da(s) linguagem(ns) de programação utilizadas; d) O processo de testes, incluindo a cobertura contra seus requisitos, comprovando a correta implementação baseada nestes requisitos, indicando critério de aceitação e o resultado esperado; e) O Processo de validação dos requisitos, arquitetura, projeto, codificação e testes, para ter certeza de que os artefatos gerados estão corretos; f) O processo de controle de configuração, incluindo a captura, fluxo e disposição dos problemas encontrados durante o desenvolvimento; e g) As ferramentas utilizadas nos processos de requisitos, arquitetura, projeto e codificação, implementação, testes, validação e controle de configuração.
RP 2	O Plano de Desenvolvimento de Software (PDS) deve ser enviado para a aprovação do certificador, na fase inicial do Produto de Software.
RP 3	O processo de requisitos deve definir os requisitos de software a partir do nível do produto que é implementado pelo software.
RP 4	Os requisitos de software devem identificar o comportamento funcional e as interfaces externas do software.
RP 5	O processo de arquitetura deve usar as saídas do processo de requisitos para desenvolver arquitetura de software, criar módulos e alocar funções expressas pelos requisitos de software.
RP 6	O processo de projeto e codificação deve usar as saídas da fase de arquitetura para decompor cada módulo em componentes de software e para definir dados e controlar o fluxo necessário entre os componentes.
RP 7	O processo de implementação deve usar as saídas do processo de projeto e codificação para gerar o código executável a partir do código-fonte e ser carregado no hardware.
RP 8	O processo de testes deve descrever os testes necessários para confirmar se a implementação foi executada corretamente.
RP 9	O processo de testes deve aplicados em três escopos diferentes para garantir a conformidade com requisitos de software: unidade, integração e teste funcional.
RP 10	O processo de testes de unidade deve confirmar que o processo de projeto e codificação implementou corretamente cada componente (unidade) do produto de software.
RP 11	O processo de testes de integração deve confirmar que os dados estão corretos e o fluxo de controle entre os módulos e componentes foram implementados corretamente.
RP 12	O processo de testes funcionais deve confirmar que os requisitos de software estão corretamente implementados no produto de software.
RP 13	O processo de validação deve confirmar a exatidão de cada processo executado. Validações são aplicadas para garantir que as saídas de cada Processo sejam corretamente definidas.
RP 14	O processo de validação de requisitos deve garantir que todos os requisitos capturados do produto e que serão aplicados ao software sejam refinados corretamente em um conjunto de requisitos de software.
RP 15	O processo de validação de arquitetura deve garantir a integridade dos módulos criados e que os requisitos de software foram alocados corretamente nos módulos.
RP 16	O processo de validação de projeto e código deve garantir: (i) correção dos dados e controle do fluxo entre os componentes; (ii) requisitos de software de baixo nível que definem lógicas e detalhes internos são corretamente refinados a partir dos requisitos de software; e (iii) o código fonte é produzido corretamente.
RP 17	O processo de validação da implementação deve garantir a exatidão do código executável produzido, garantindo que todos os componentes estejam corretamente vinculados e construído.
RP 18	O processo de validação da fase de verificação deve garantir que os casos de teste, procedimentos e resultados confirmem que o software é implementado conforme especificado pelos requisitos de software, arquitetura, projeto e código.
RP 19	O processo de controle de configuração de software deve identificar e controlar todos os artefatos gerados durante o desenvolvimento do software seguindo todos os processos previstos nos demais Requisitos de Processos, incluindo a captura de problemas e suas disposições.
RP 20	Ao final do desenvolvimento de software, e suas modificações posteriores, a contratada deve gerar um Sumário de Configuração de Software (SCS), indicando a versão de software entregue, suas instruções de instalações e suas possíveis limitações existentes na ocasião da entrega.
RP 21	Ao final do desenvolvimento de cada versão, a contratada deve entregar à contratante uma versão do produto de software desenvolvida seguindo o processo de desenvolvimento que atendeu aos Requisitos de Processos definidos neste trabalho.
RP 22	A contratada deve entregar para a autoridade responsável pela aprovação da nova versão: o produto de software a ser instalado e o Sumário de Configuração de Software (SCS), a cada versão a ser instalada no SMS.

Estes ciclos de vida encontram-se sintetizados em [6]. A Fig. 2 apresenta a integração dos 22 Requisitos de Processos com os principais ciclos de vida e envolvendo os três papéis principais na certificação militar.

Além dos trabalhos apresentados na seção VI, que suportaram a criação dos Requisitos de Processos, os demais trabalhos referenciados na Tabela 2 representam os levantamentos de revisão bibliográfica e em normas e modelos utilizados conforme identificado nos passos 1 e 2 da Fig. 1. Esses trabalhos suportam a necessidade destes requisitos de processos identificados.

TABELA II. MAPEAMENTO DOS REQUISITOS NA LITERATURA

Processo	*Trabalhos Identificados na Revisão Bibliográfica*	*Normas e Modelos*
Planejamento (RP 1 e 2) e fechamento (RP 21 e 22)	Nenhum	[4]
Requisitos (RP 3 e 4)	[22][23][24][25][30] [32][33][34][35][36]	[4][7][9][10][48]
Arquitetura (RP 5), projeto e codificação (RP 6) e implementação (RP 7)	[21][22][37][38][39] [40]	[4][7][9][10][48]
Testes (RP 8 a 12) e validação (RP 13 a 18)	[23][24][41][42][43]	[4][7][9][10][48]
Controle de configuração (RP 19 e 20)	[25][44][45][46][47]	[4][7][9][10][48]

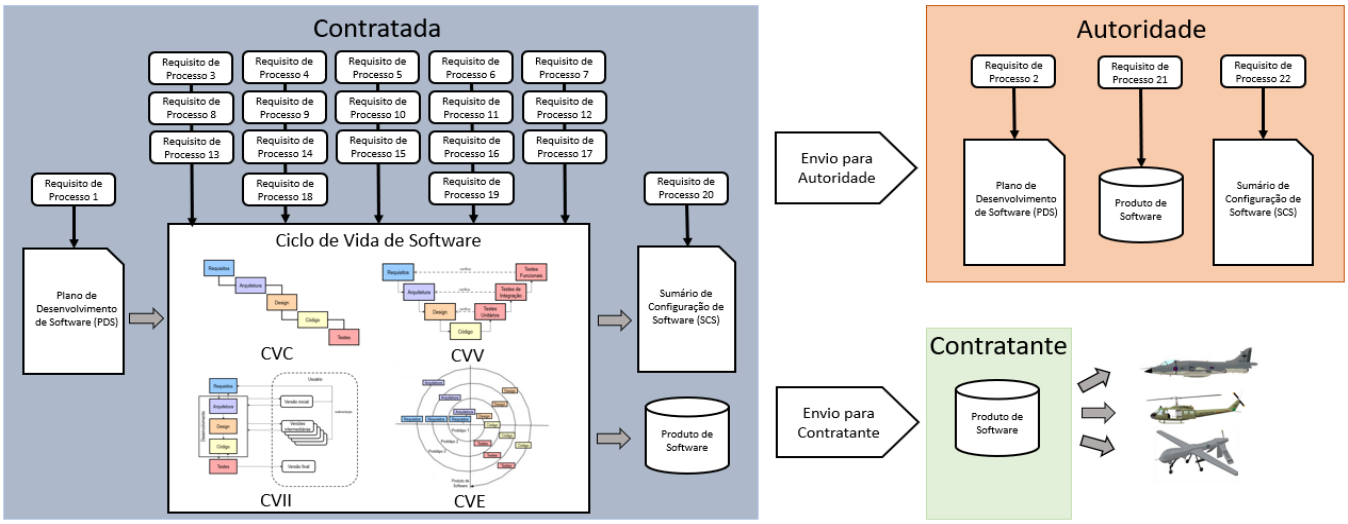

Fig. 2. Integração dos 22 Requisitos de Processos e Ciclos de Vida

VIII. CONCLUSÃO E TRABALHOS FUTUROS

Este artigo apresentou levantamento dos requisitos para a certificação de SMS, visando a garantia de qualidade do produto de software com segurança e cumprimento da missão atribuída. Os 22 Requisitos de Processos são considerados pelos autores deste trabalho como um conjunto mínimo de requisitos que devem ser integrados nos processos definidos pela empresa contratada para o desenvolvimento de software militar.

Este trabalho representa uma proposta, em fase de elaboração e com necessidade de futura aplicação em desenvolvimentos de SMS, eliminando assim a necessidade de prescrição de uma norma ou modelo de maturidade específico.

Fica pendente como um trabalho futuro a execução dos passos 4 e 5 da Fig. 1, pois apenas com a conclusão desses cinco passos e aplicação dos 22 Requisitos de Processos, os autores serão capazes de comprovar a efetividade dessa nova abordagem. Como é um trabalho em andamento, o uso efetivo dos Requisitos de Processos identificados na Tabela 2, ainda se encontram em validação pelos autores deste trabalho.

Este trabalho representa um estudo, em andamento no Instituto de Fomento e Coordenação Industrial (IFI), responsável pelas certificações militares de produtos do Comando da Aeronáutica e conta com a colaboração do Instituto Tecnológico de Aeronáutica (ITA). Ambos os institutos (ITA e IFI) são ligados ao Departamento de Ciência e Tecnologia Aeroespacial (DCTA), Comando da Aeronáutica (COMAER) e Ministério da Defesa (MD).

REFERÊNCIAS

[1] J.C. Marques, L.A.V. Dias, “Mitigação de riscos de certificação civil em um processo de certificação militar: uma abordagem para software embarcado”, XII Simpósio de Aplicações Operacionais em Área de Defesa, 2010.

[2] Ministério da Defesa – Comando da Aeronáutica, “Regulamento de Aeronavegabilidade Militar – ICA 57-21 Procedimentos para Certificação de Produto Aeronáutico”, 2017.

[3] J.C. Marques, A.M. Cunha, “Especificação de Requisitos de Software: Um Modelo Ágil para Ambientes Regulados”. 1ª ed. Brasil: Novas Edições Acadêmicas, 2017.

[4] Radio Technical Commission for Aeronautics, “DO-178C - Software Considerations in Airborne Systems and Equipment Certification”. Estados Unidos, 2011.

[5] R.C. Martins, S.R.M. Pellegrino, J. Santellano, “Tratamento de dead codes em software de uso aeronáutico”, pp: 718-725, 9th Conference on Dynamics, Controls and their Applications, 2010.

[6] J.C. Marques, A.M. Cunha, “Tailoring traditional software life cycles to ensure compliance of RTCA DO-178C and DO-331 with model-driven design, pp 1-8, 37th IEEE/AIAA Digital Avionics System Conference, 2018.

[7] International Standardization Organization, “ISO/IEC/IEEE 12207:2017 Systems and software engineering — Software lifecycle processes”, Estados Unidos, 2017.

[8] International Electrotechnical Commission, “IEC61508-3:2010 Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related System - Software Requirements”, Estados Unidos, 2010.

[9] M.B. Chrissis, M. Konrad, S. Shrum, “CMMI for Development: Guidelines for Process Integration and Product Improvement.” 3a. ed. Estados Unidos: Addison-Wesley Professional, 2011.

[10] Softex, “MPS.BR - Melhoria de Processo do Software Brasileiro”, Brasil, 2012.

[11] International Standardization Organization, “ISO 15504:2012 Information Technology - Process Assessment Systems and software engineering — Software lifecycle processes”, Estados Unidos, 2017.

[12] P. Chen, M. Unewisse, “A systems thinking approach to engineering challenges of military systems-of-systems”, Joint & Operations Analysis Division Defence Science and Technology Group, Australia 2016.

[13] T. Kim, S. Park, T. Lee, “Applying software reliability engineering process to software development in Korea defense industry”, 2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW), França, 2017.

[14] B. Çiflikli, O. Özcan, A.B. Uysal, “Model driven software development for military command and control systems”, 2012 IEEE 36th International Conference on Computer Software and Applications, Turquia, 2012.

[15] W. Li, X. Yang, J. Huang, Y. Zang, “Research on integrated evaluation of military software supportability based on FAHP”, 11th International

Conference on Reliability, Maintainability and Safety (ICRMS), China, 2016.

[16] Q. Zhou, B. Liu, Z. Yu, X. Xing, "The process of requirement Analysis about military software system testing", 9th International Conference on Reliability, Maintainability and Safety, China, 2011.

[17] L. Benedicenti, A. Messina, A. Sillitti, "IAgile: mission critical military software development", 2017 International Conference on High Performance Computing & Simulation, Italia, 2017.

[18] P. Tsoucamoto, P., J.C. Marques, A.M Cunha, "Uma abordagem de desenvolvimento inspirada em práticas ágeis para sistemas de softwares embarcados militares", XIX Simpósio de Aplicações Operacionais em Áreas de Defesa (SIGE), Brasil, 2017).

[19] R. Oshana, M. Kraeling, "Software Engineering for Embedded Systems: Methods, Practical Techniques and Applications", 1a. ed., Estados Unidos:Elsevier, 2013.

[20] D. Neiva, F.C. Almeida, E.S. Almeida, S.L. Meira, "A requirements engineering process for software product lines.", 2010 IEEE International Conference on Information Reuse & Integration, Estados Unidos, 2010.

[21] S. Stevanetic, U. Zdun, U., "Empirical study on the effect of a software architecture representation's abstraction level on the architecture-level software understanding, International Conference on Quality Software, Estados Unidos, 2014.

[22] A. Contan, C. Dehelean, L. Miclea, "Applying coding systems in the process of testing software applications", 14th International Conference on Engineering of Modern Electric Systems (EMES). Romênia (2017).

[23] Dudila, R., Letia, I. A.: Towards combining functional requirements tests and unit tests as a preventive practice against software defects. In: 2013 IEEE 9th International Conference on Intelligent Computer Communication and Processing (ICCP), Romênia, 2013.

[24] J.M. Reddy, S.V.A.V. Prasad, "The role of verification and validation in software testing", 3rd International Conference on Computing for Sustainable Global Development (INDIACom), India, 2016.

[25] J. Wu, H. Hai Liu, "The study of configuration software and management information systems integration", 2010 International Conference On Computer Design and Applications, China, 2010.

[26] R. Pressman, B.R. Maxim, "Engenharia de Software: Uma Abordagem Profissional", 8ª ed. Brasil: Bookman, 2016.

[27] I. Sommerville, "Engenharia de Software". 10ª ed. Brasil: Pearson, 2019.

[28] D.G. Firesmith, "Engineering safety and security-related requirements for software-intensive systems: tutorial summary", ACM/IEEE 32nd International Conference on Software Engineering, Africa do Sul, 2010.

[29] D. Pandey, U. Suman, A.K. Ramani, "An effective requirement engineering process model for software development and requirements management", 2010 International Conference on Advances in Recent Technologies in Communication and Computing, India, 2010.

[30] I. Basharat, M. Fatima, R. Nisa, R. Hashim, A. Khanum, "Requirements engineering practices in small and medium software companies: An empirical study", 2013 Science and Information Conference, Reino Unido, 2013.

[31] R. Sinha, S. Patil, C. Pang, V. Vyatkin, B. Dowdeswell, "Requirements engineering of industrial automation systems: Adapting the CESAR requirements meta model for safety-critical smart grid software", 41st Annual Conference of the IEEE Industrial Electronics Society, Japão, 2015.

[32] K. Meridji, G. Issa, "A development approach of software requirements for renewable energy applications using fundamental principles of software engineering", 1st International Conference & Exhibition on the Applications of Information Technology to Renewable Energy Processes and Systems, Jordania, 2013.

[33] M. Hinchey, "Rethinking Functional Requirements: A Novel Approach Categorizing System and Software Requirements: Software Technology: 10 Years of Innovation in IEEE Computer". Estados Unidos: Wiley-IEEE Press eBook Chapters, 2018.

[34] A.Y. Zalozhnev, "Big banks' management control systems' software: Architecture. General requirements and functional components", 10th International Conference Management of Large-Scale System Development (MLSD), Russia, 2017.

[35] X. Lian, L. Zhang, "Optimized feature selection towards functional and non-functional requirements in software product lines", 22nd International Conference on Software Analysis, Evolution, and Reengineering (SANER), Canadá, 2015.

[36] V.S. Sharma, S. Sarkar, K. Verma, A. Panayappan, A. Kass, "Extracting high-level functional design from software requirements", Asia-Pacific Software Engineering Conference, Malásia, 2009.

[37] H. Gomaa, "Designing software product lines with UML 2.0: from use cases to pattern-based software architectures", 10th International Software Product Line Conference (SPLC'06), Estados Unidos, 2006.

[38] A. Amirat, "Generic model for software architecture evolution", 2012 International Conference on Advanced Computer Science Applications and Technologies (ACSAT), Malásia, 2012.

[39] I. Park, H. Lee, S. Lee, D. Kim, W.G. Lim, "Source code optimization for embedded processing software of tactical communication system", 7th International Conference on Computing and Convergence Technology (ICCCT), Coréia do Sul, 2012.

[40] M. Bernhart, S. Reiterer, K. Matt, A. Mauczka, T. Grechenig, "A task-based code review process and tool to comply with the DO-278/ED-109 standard for air traffic management software development: an industrial case study", 13th International Symposium on High-Assurance Systems Engineering, Estados Unidos, 2011.

[41] W. Ahmad, U. Qamar, S. Hassan, "Analyzing different validation and verification techniques for safety critical software systems", 6th IEEE International Conference on Software Engineering and Service Science (ICSESS), China, 2015.

[42] E. Jharko, "Verification and software quality assurance for nuclear power engineering", 11th International Conference Management of large-scale system development, Rússia, 2018.

[43] M. Dodds, S. Magill, S., A. Tomb, "Continuous verification of critical software", 2018 IEEE Cybersecurity Development (SecDev), Estados Unidos, 2018.

[44] T.N. Nguyen, "A unified model for product data management and software configuration management", 21st IEEE/ACM International Conference on Automated Software Engineering (ASE'06), Japão, 2006.

[45] T. Kim, S. Kang, J. Lee, "Effects of variable part auto configuration and management for software product line", 42nd Annual Computer Software and Applications Conference (COMPSAC), Japão, 2018.

[46] S. Pei, D. Chen, "The implementing of software configuration management based on CMM", 5th International Conference on Wireless Communications, Networking and Mobile Computing, China, 2009.

[47] S.S.M. Fauzi, N. Ramli, "Software configuration management: a result from the assessment and its recommendation" 2009 International Conference on Information Management and Engineering, Malásia, 2009.

[48] Departament of Defense "MIL-STD-498 Software Development and Documentation". Estados Unidos, 1994.