LP-SAR: Linhas de Processo para Software em Ambientes Regulados

Johnny Marques
johnny@ita.br
Instituto Tecnológico de Aeronáutica
São José dos Campos, Brazil

Sarasuaty Yelisetty
sara.mhy@gmail.com
Instituto Tecnológico de Aeronáutica
São José dos Campos, Brazil

RESUMO

Com grande notoriedade nos últimos tempos, os ambientes regulados por normas, tradicionalmente os ambientes *safety-critical*, possuem disciplina de execução. Historicamente, normas das diversas tecnologias disponíveis, incluindo a engenharia de software, foram surgindo ao longo dos últimos 30 anos. Percebe-se na revisão destas normas um conjunto de interseção bem amplo entre seus processos, atividades, entradas e saídas. O objetivo deste artigo curto é apresentar um modelo geral com linhas de processo, em elaboração inicial, para o desenvolvimento de software, visando a aplicação multidomínios em ambientes regulados. Este trabalho encontra-se em andamento na Divisão de Ciência da Computação do Instituto Tecnológico de Aeronáutica.

KEYWORDS

software, *safety*, processos, requisito, teste

ACM Reference Format:
Johnny Marques and Sarasuaty Yelisetty. 2019. LP-SAR: Linhas de Processo para Software em Ambientes Regulados. In *BWare 2019, 23 a 27 de Setembro de 2019, Salvador, BA, Brasil.* ACM, New York, NY, USA, 4 pages. https://doi.org/TBD

1 INTRODUÇÃO

Segundo Pressman & Maxim (2016) [13], a Engenharia de Software é a área responsável pelo estabelecimento de técnicas e práticas para o desenvolvimento de software, cobrindo uma ampla quantidade de aplicações e diferentes tipos de dispositivos.

Diversas metodologias de desenvolvimento de software são compostas por sequências de fases e atividades, com entradas, saídas, regras, papéis, técnicas, notações, ferramentas e documentação. Estas metodologias surgiram com o objetivo de se prestar mais atenção aos meios de viabilização da Engenharia de software e menos à programação e implementação de seus produtos.

Vuori (2011) [16] considera que se um erro de software causar potencialmente a perda de vidas humanas ou outras consequências catastróficas, este é denominado como um software crítico quanto à segurança (software *safety-critical*).

Tipicamente os software *safety-critical* são desenvolvidos em ambientes regulados por normas e padrões. Exemplos deste tipo de software são encontrados em domínios, tais como: Aeronáutico, Médico, Ferroviário, Nuclear entre outros. Nesta pesquisa, os software destes domínios são tratados como software em ambientes regulados.

De acordo com Clements & Northrop (2015) [3], uma linha de produto de software é uma técnica de produção baseada em outras engenharias. Seu intuito concentra-se em desenvolver uma mesma família de produtos de software com partes e recursos comuns. Por outro lado, Carvalho et al. (2014) [6] define que as Linhas de Processo de Software (LPS) representam um conjunto de processos semelhantes dentro de um domínio específico ou para um propósito específico, com características comuns e construído com base em recursos de processo comuns e reutilizáveis.

O objetivo deste artigo curto é *apresentar um modelo geral com Linhas de Processo de Software (LPS), em elaboração inicial, para o desenvolvimento de software, visando a aplicação multidomínios em ambientes regulados.*

Além desta seção 1, o trabalho ontem outras 5. A seção 2 apresenta uma breve fundamentação teórica. A seção 3 apresenta o problema de pesquisa e a metodologia escolhida pelos autores. Já na seção 4 é apresentada uma visão do estrutural do projeto LP-SAR. A seção 5 apresenta exemplos iniciais já obtidos. A conclusão e trabalhos futuros é apresentada na seção 6.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Visão Geral sobre Software em Ambientes Regulados

Os ambientes regulados são aqueles que trazem impactos à sociedade em geral e, por isto, precisam de padrões que legislem sobre os produtos e serviços entregues por empresas e organizações envolvidas. Existe a expectativa da sociedade de receber serviços e produtos seguros e confiáveis. Em todos os diversos ambientes regulados, como: aeronáutico, ferroviário, automotivo, nuclear, médico, militar, entre outros, existem padrões que abrangem diversas tecnologias, incluindo o desenvolvimento de software. Como consequência direta, existem normas que regulamentam as exigências de demonstração que um produto de software é seguro e confiável para se operar neste tipo de ambiente.

Munch et al. (2012) [10] consideram que o número de organizações que precisam verificar aderência aos padrões regulatórios vêm aumentando. Muitos destes regulamentos, que se apresentam como normas, requerem a presença de processos de desenvolvimento de software explícitos. Neste aspecto, as atividades realizadas devem apresentar repetibilidade e rastreabilidade dentro do processo de desenvolvimento de software proposto.

BWare 2019, 23 a 27 de Setembro de 2019, Salvador, BA, Brasil

ACM ISBN TBD.
https://doi.org/TBD

Essas normas possuem objetivos ou atividades que precisam ser satisfeitas para que os produtos de software e/ou seus fabricantes sejam aprovados (certificados) para operação em seu ambiente de utilização. Agências reguladoras, ou outras entidades, normalmente, exigem a aderência às normas e padrões estabelecidos, como na IEC 62279 [5] para o domínio ferroviário e nas demais apresentadas na Tabela 1.

Muitas destas normas são estratificadas em níveis de criticalidade, como a RTCA DO-178C que possui 5 níveis: A (mais crítico), B, C, D e E (sem impacto em segurança de voo). Já a IEC 62304 possui três níveis, denominados como classes: C (mais crítico), B e A (sem risco ao paciente). A IEC 61508-3 e a IEC 62279 também possuem níveis de criticalidade.

Tabela 1: Normas de Software em Ambientes Regulados

Domínio	Norma de Software	Ano
Aeronáutico	RTCA DO-178C [14]	2011
Aeroespacial	ECSS-E-ST-40C [8]	2010
Ferroviário	IEC 62279 [5]	2015
Médico	IEC 62304 [11]	2015
Militar	ISO 12207 [12]	2017
Nuclear	IAEA SSG-39 [1]	2016
Químico e Automotivo	IEC 61508-3 [4]	2010

2.2 Visão Geral sobre Sistemas Especialistas Baseados em Regras

De acordo com Ryan (2017) [15], os Sistemas Especialistas Baseados em Regras são tipicamente compostos de:

- Uma Memória de Trabalho (MT): conjunto de dados que representa estado corrente do mundo;
- Um Conjunto de Regras (CR): compõe a base de conhecimento e é utilizada pela máquina de inferência para processar os dados de entrada. Um CR segue a forma C (Condição) → A (Ação) logo cada Regra fica da forma: *if < Condição em MT> then Ação*; e
- Uma Máquina de Inferência (MI): Infere conclusões a partir de fatos (dados de entrada) e da base de conhecimento.

A Máquina de Inferência realiza o raciocínio lógico que permite que o sistema especialista resolva um problema. Esse é o processo de vinculação das regras, presentes na base de conhecimento, aos fatos contidos na base de dados [15]. A Figura 1 apresenta a estrutura básica de um Sistema Especialista Baseado em Regras.

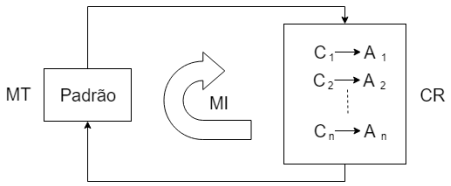

Figura 1: Estrutura Básica de um Sistema Especialista Baseado em Regras

A escolha pelo uso de Sistema Especialista Baseado em Regras se dá justamente porque as normas de software em ambientes regulados apresenta um conjunto de atividades e objetivos que precisam ser satisfeitos e demonstrados durante o processo de desenvolvimento de software. Estas atividades e objetivos normalmente apresentam fortes vínculos entre si. Em geral estas normas podem alimentar os três componentes MT, CR e MI.

2.3 Visão Geral sobre Linhas de Processo de Software

De acordo com Carvalho et al. (2014) [7], objetivo final de melhorar a qualidade do software é a principal motivação para a definição do processo de software, com base na premissa de que o processo pelo qual um software é desenvolvido influencia fortemente a qualidade desse software. No entanto, a definição do processo é uma atividade complexa, que requer experiência e conhecimento de uma variedade de disciplinas de engenharia de software. Além disso, definir processos de software a partir do zero para cada novo projeto cria altos riscos e requer uma quantidade significativa de esforço e tempo.

Segundo Armbrust et al. (2008) [2], a diferença fundamental entre Linhas de Processo de Software (LPS) e paradigmas tradicionais para adaptação de processos é que, utilizando LPS, os processos de software são preparados de maneira proativa para uma variedade de necessidades futuras, que são incorporadas na própria estrutura da LPS. Isso permite a reutilização em larga escala de processos.

3 PROBLEMA DE PESQUISA E METODOLOGIA

De acordo com Hawkins et al.(2013) [9], existem princípios comuns entre as normas de ambientes regulados, considerados como um núcleo central imutável, onde se espera a possibilidade de estabelecimento de uma linha única de processos, que podem ser utilizados e reusados, parcialmente ou integralmente, em diversos domínios de ambiente regulados. Assim, o autor desta pesquisa define o termo "multidomínios" como um conjunto com dois ou mais domínios de ambientes regulados. Desta forma, uma solução que atende a um ambiente regulável pode ser reutilizável integralmente ou com alguma adaptação para o atendimento de outros domínios. A Figura 2 ilustra o conceito multidomínios estabelecido nesta pesquisa.

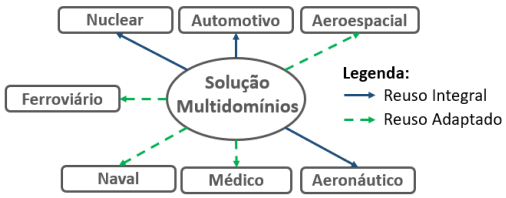

Figura 2: Visão do Autor sobre o Conceito Multidomínios

O problema abordado neste projeto de pesquisa relaciona-se com o fato de que, atualmente, não existe um padrão único para software em ambientes regulados que seja e aplicável a mais de um domínio. O problema a ser endereçado neste projeto de pesquisa consiste em

"Como criar um modelo geral que possa ser adaptado e configurado, gerando várias Linhas de Processo e atendendo multidomínios?".

Ter um modelo geral pode ser útil para empresas de desenvolvimento de software embarcado, principalmente crítico, possibilitando ter visão da vários ambientes regulados, compartilhando do máximo de reuso de processos, atividades e ferramentas, com adaptações para cada linha de processo, quando necessário.

4 VISÃO ESTRUTURAL DO PROJETO LP-SAR

O Projeto LP-SAR encontra-se estruturado em três camadas. A Camada 1 representa a estrutura em *Processos*. Já a Camada 2 representa a estrutura em *Atividades*, que representam um detalhamento de cada processo. Finalmente, a Camada 3 representa a estrutura em *Tarefas* que é o nível operacional para a execução. As *Tarefas* consomem *Entradas* e produzem *Saídas*. A Figura 3 apresenta uma organização espacial desta estrutura.

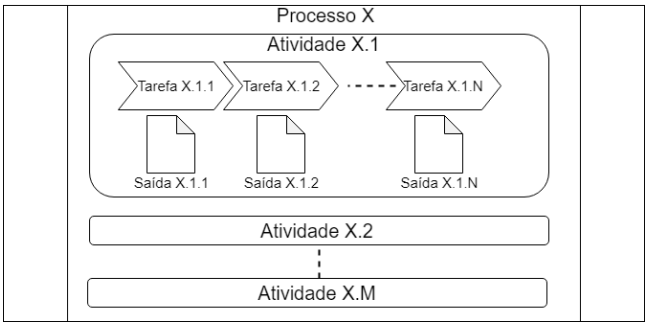

Figura 3: Organização da Estrutura de Definição

Cada *Linha de Processo* tem uma estrutura preliminar em 5 processos, já identificados e que serão configurados de acordo com as regras recebidas pelo *Sistema Especialista*.

Os seguintes processos encontram-se em definição para viabilizar a configuração em cada *Linha de Processo*, sendo eles:

- *Processo 1*: Definição dos Requisitos de Software;
- *Processo 2*: Definição do Design e Codificação;
- *Processo 3*: Testes e Verificação;
- *Processo 4*: Gerenciamento de Configuração; e
- *Processo 5*: Revisão e Validação.

Os cinco processos funcionarão em ciclos incrementais. A Figura 4 apresenta a estrutura de sequência dos 5 processos a serem definidos.

A visão por domínio, ou seja, em cada *Linha de Processo*, pode necessitar de adaptações que envolvem: inclusões, modificações ou eliminações de partes do modelo geral. A Figura 5 apresenta uma representação deste conceito.

Esta proposta envolve a criação de quatro entidades:

- Modelo Geral;
- Configuração;
- Sistema Especialista; e
- Linhas de Processo.

A entidade *Modelo Geral* envolve a investigação dos processos existentes nas normas apresentadas na Tabela 1, descobrindo as atividades, entradas e saídas requeridas por estas normas. A entidade *Configuração* envolve a comfiguração de domínio, identificando que norma deverá ser satisfeita pelo sistema especialista em regras. A entidade *Sistema Especialista* envolve a seleção de regras para estabelecimento da *Linhas de Processo Específica* do domímio regulado. A entidade *Linhas de Processo* contém todas as *Linhas de Processo Específicas* aplicáveis para os diversos ambientes regulados e normas apresentados na Tabela 1.

5 EXEMPLOS INICIAIS

Esta seção apresenta dois exemplos iniciais já coletados com o uso das normas RTCA DO-178C [14] e IEC 62304 [5].

5.1 Exemplo 1: RTCA-DO-178C

Como exemplo da construção de regras baseadas na Memória de Trabalho existente na RTCA DO-178C, foi utilizado o *Processo 1: Definição dos Requisitos de Software* e a sua atividade *P1_At1: Construção dos Requisitos de Software*.

Essa atividade quando configurada com a *Configuração de Domínio: RTCA_DO_178C*, para o *Nível Associado: A*, seleciona a Regra *IF obj_A2_1=TRUE, THEN T1=TRUE*, configurando assim a tarefa *T1: Definir os Requisitos Funcionais de Software*. A Tabela 2 apresenta uma síntese deste Exemplo 1.

5.2 Exemplo 2: IEC 62304

Como um segundo exemplo, os autores apresentam a construção de regras baseadas na Memória de Trabalho existente na IEC 62304. Foi utilizado o *Processo 1: Definição dos Requisitos de Software* e a sua atividade *P1_At1: Construção dos Requisitos de Software*.

Essa atividade quando configurada com a *Configuração de Domínio: IEC_62304*, para o *Nível Associado: C*, seleciona a Regra *IF sec_5.2.2_a=TRUE, THEN T1=TRUE*, configurando assim a tarefa *T1: Definir os Requisitos Funcionais de Software*. A Tabela 2 apresenta uma síntese deste Exemplo 2.

Assim pode-se perceber que a mesmta tarefa T1 atende respectivamente a RTCA DO-178C e a IEC 62304. Sendo esta habilitada para a Linha de Processo de cada uma destas normas, quando aplicável.

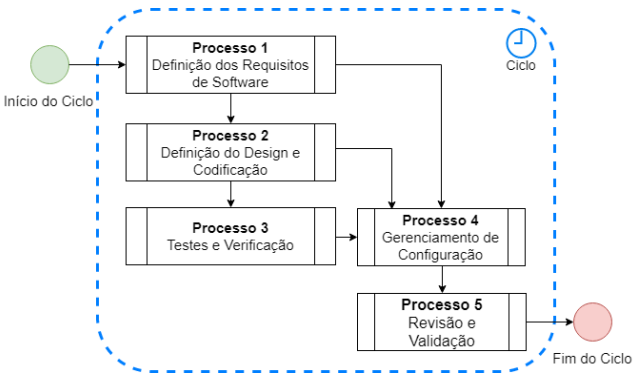

Figura 4: Sequência de Processos em Definição

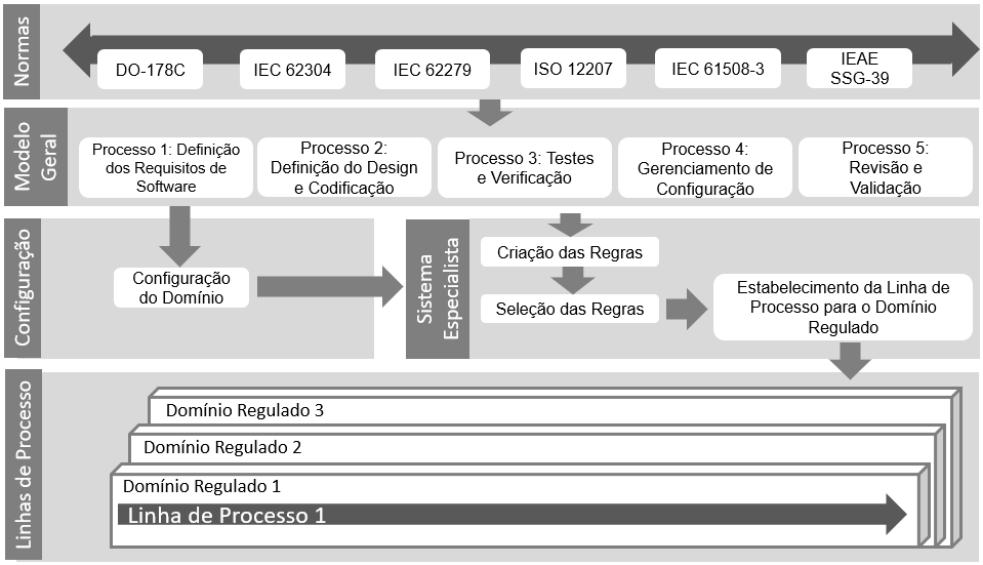

Figura 5: Normas, Modelo Geral, Configuração, Sistema Especialista e Linhas de Processo

Tabela 2: Exemplo 1: Seleção e Configuração das Regras na RTCA DO-178C [14]

Atributo	Descrição Ex1	Descricão Ex2
Norma	RTCA_DO_178C	IEC_62304
Nível	A	C
Processo	1	1
Atividade	P1_At1	P1_At1
Regra	IF obj_A2_1=TRUE, THEN T1=TRUE	IF sec_5.2.2_a=TRUE, THEN T1=TRUE

6 CONCLUSÕES PRELIMINARES E PRÓXIMOS PASSOS

O objetivo deste artigo curto foi o de apresentar um modelo geral com linhas de processo, em elaboração inicial, para o desenvolvimento de software, visando a aplicação multidomínios em ambientes regulados. Este trabalho encontra-se em andamento na Divisão de Ciência da Computação do Instituto Tecnológico de Aeronáutica.

Os autores apresentaram uma visão geral sobre a fundamentação teórica necessária para o projeto LP-SAR, envolvento: Software em Ambientes Regulados, Sistemas Especialistas Baseados em Regras e Linhas de Processo de Software.

As Linhas de Processo Específicas encontram-se em construção e são orientadas por uma visão por domínio, ou seja, em cada uma delas pode necessitar de adaptações que envolvem: inclusões, modificações ou eliminações de partes do modelo geral. Uma configuração por domínio, que permite a seleção de Regras aplicáveis, viabilizará estas necessidades de adaptação.

Os próximos passos desta pesquisa se concentram em fazer a definição completa das Regras extraídas da Memória de Trabalho existente em cada uma das normas apresentadas na Tabela 1, viabilizando assim a configuração das atividades, tarefas, entradas e saídas aplicáveis para cada Linha de Processo.

REFERÊNCIAS

[1] International Atomic Energy Agency. 2016. Specific Safety Guidance SSG-39: Design of Instrumentation and Control Systems for Nuclear Power Plants.

[2] Ove Armbrust, Masafumi Katahira, Yuko Miyamoto, Jorgen Munch, Haruka Nakao, and Alexis Ocampo. 2008. Scoping Software Process Models - Initial Concepts and Experience from Defining Space Standards. *International Conference on Software Process ICSP 2008: Making Globally Distributed Software Development a Success Story* (2008), Pages 160–172.

[3] Paul Clements and Linda Northrop. 2015. *Test-driven Development: A Practical Guide.* Addison-Wesley.

[4] International Electrotechnical Commission. 2010. IEC61508-3 Functional Safety of Electrical/Electronic/Programmable Electronic Safety Related SySyste - Software Requirements.

[5] International Electrotechnical Commission. 2015. IEC 62279 Railway applications - Communication, signalling and processing systems - Software for railway control and protection systems.

[6] Daniel Dias de Carvalho, Larissa Fernandes Chagas, Adailton Magalhaes Lima, and Carla Alessandra Lima Reis. 2014. Software Process Lines: A Systematic Literature Review. *International Conference on Software Process Improvement and Capability Determination SPICE 2014: Software Process Improvement and Capability Determination* (2014), Pages 118–130.

[7] Daniel Dias de Carvalho, Larissa Fernandes Chagas, Adailton Magalhaes Lima, and Carla Alessandra Lima Reis. 2014. Software Process Lines: A Systematic Literature Review. *International Conference on Software Process Improvement and Capability Determination SPICE 2014: Software Process Improvement and Capability Determination* (2014), Pages 118–130.

[8] European Cooperation for Space Stardardization. 2009. ECSS-E-ST-40C - Space Engineering/Software.

[9] Richard Hawkins, Ibrahim Habli, and Tim Kelly. 2013. The Principles of Software Safety Assurance. (August 2013).

[10] Jurgen Munch, Ove Armbrunt, Martin Kowalczyk, and Martin Soto. 2012. *Software Process Definition and Management.* Springer-Verlag, Berlim, Germany.

[11] International Standardization Organization. 2015. IEC/ISO 62304 Medical Device Software - Software Life-Cycle Processes.

[12] International Standard Organization. 2017. ISO/IEC 12207 Systems and software engineering -Software life cycle processes.

[13] Roger Pressman and Bruce Maxim. 2014. *Software Engineering: A Practitioner's Approach.* Pearson Education, Estados Unidos da América.

[14] RTCA. 2011. DO-178C Software Considerations in Airborne Systems and Equipment Certification.

[15] Darrel Ryan. 2017. *Expert Systems: Design, Applications and Technology.*

[16] Matti Vuori. 2011. *Agile Development of Safety-Critical Software.* Technical Report. Tampere University of Technology.