X Workshop em Engenharia e Tecnologia Espaciais
7 a 9 de agosto de 2019

Método de Identificação de Requisitos de um Software de OBDH de Nanosatélites a Partir de Serviços PUS

PADILHA, J. J. [1], AMBROSIO, A. M. [2], MATTIELLO-FRANCISCO, F.[3]

[1]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

Aluna de Doutorado do curso de Engenharia e Tecnologia Espaciais - Área de Concentração em Engenharia e Gerenciamento de Sistemas Espaciais.

[1]Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

Doutorando na PG-ETE e a Área de Concentração em Engenharia e Gerenciamento Espaciais.

[2] DSE, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

[3] COCRE, Instituto Nacional de Pesquisas Espaciais, São José dos Campos, SP, Brasil

juliana.padilha@gmail.com

Resumo. *O objetivo da tese é propor uma estratégia para gerar casos de testes para software de bordo de nanosatélites com uma abordagem baseada em redução de esforço de teste. Neste artigo, propomos um método auxiliar chamado Gerador de Tabela de Transição de Estados, que é parte da estratégia da tese. Este método gera uma tabela de transições de estados para representar os requisitos do software a ser testado. A tabela é construída a partir de tabelas de referência que representam os requisitos de um serviço da norma PUS e baseia-se na metodologia Conformance and Fault Injection e em alguns conceitos de Linha de Produtos de Software para mapeamento de características. O resultado desse método, será utilizado para a geração de modelos (em particular, máquinas de estados finitos) os quais serão usados para geração de casos de testes.*

Palavras-chave: Tabela de transição de estados; Máquinas de estados finitos; Testes; Nanosatélites; PUS.

1. Introdução

Garantir a qualidade de sistemas na área espacial é muito importante, pois pode evitar a perda da missão. Isso acontece, inclusive, para os nanosatélites que são voltados para habilitar estudantes a projetar, construir, testar e operar satélites. Porém, são poucos os documentos existentes para orientar o desenvolvimento e os testes para a verificação e

validação de software embarcado (OBDH, do inglês OnBoard Data Handling) neste tipo de satélites. Algumas empresas utilizam processos particulares e outras utilizam os padrões voltados a satélites tradicionais [Scholz 2017].

Considerando que a (i) realização de testes consome de 30% a 50% dos recursos do projeto de software [De Padua Paula Filho 2003], que (ii) software de bordo para nanosatélites são tão complexos quanto o software de bordo de satélites tradicionais e (iii) o tempo e o custo de desenvolvimento e verificação devem ser menores do que o dos satélites tradicionais, essa pesquisa tem o objetivo de buscar a redução do esforço de teste no contexto descrito.

A norma ECSS-E-ST-70-41C [European Cooperation For Space Standardization 2016], ou também chamada PUS, trata da utilização de pacotes de telecomando e pacotes de telemetria para fins de monitoramento e controle remoto de subsistemas e cargas úteis. Elas descrevem um conjunto de serviços típicos de software a bordo de satélites. Existem estudos que mostram que o comportamento desses serviços podem ser representados através de modelos de estados. [Ambrosio and Martins 2006] [Greghi et al. 2018]

Seguindo a ideia de organização do software de bordo em serviços e a modelagem destes serviços em modelos de estados visando geração de casos de teste, no Projeto Qualidade de Software Embarcado em Aplicações Espaciais (QSEE), o Software Payload Data Handling Computer (SWPDC) foi mapeado em serviços e modelos de estados construídos visando a geração de Teste de Aceitação, uma vez que o SWPDC foi desenvolvido por terceiros. Neste projeto, aplicou-se a metodologia de teste baseado em modelos, chamada Conformance and Fault Injection (CoFI), com resultados promissores [Ambrosio 2007]. As funções desse software foram identificadas e mapeadas a um subconjunto dos serviços da norma PUS.

Considerando que o software de bordo de um nanosatélite também deve possuir um subconjunto dos serviços e que a norma PUS especifica serviços de software de bordo, tem se algumas semelhanças entre as duas situações. Estes aspectos, inspiram a exploração de uma abordagem de teste que inclua mapeamento de serviços de software de bordo de nanosatélites, sua representação em modelos e a indicação de reuso (dos modelos dos serviços) entre nanosatélites e consequentemente levando a proposição de uma abordagem de teste para Verificação e Validação de Software Embarcado em Nanosatélites.

No desenvolvimento desta pesquisa propusemos a criação de um método que gera uma tabela de transição de estados a partir de uma Tabela de Transição de Estados Referência (TTER), chamado Gerador de Tabela de Transição de Estados (GeTTE). Este método deverá produzir uma tabela de transição de estados que represente os requisitos do OBDH do nanosatélite em teste que servirá para a geração dos casos de testes. O método apresentado aqui fará parte da estratégia que está sendo desenvolvida que visa a redução do esforço de testes para software embarcados em nanosatélites, através da reutilização de modelos de serviços para a geração de testes entre diversos nanosatélites.

2. Metodologia

A aplicação da metodologia COFI mostra a viabilidade da modelagem de serviços da norma PUS em máquinas de estado finitos [Ambrosio and Martins 2006]. Estas ideias foram usadas para a geração da TTER, tendo como entrada um dos serviços da norma PUS. Nesta etapa não seria necessário o uso da COFI, porém alguns dos motivos de utiliza-la se deveu ao fato

dela ser uma metodologia que (i) usa técnica de testes baseados em modelos para a geração de máquinas de estados finitos, (ii) tem o objetivo de reuso de modelos e (iii) permite a geração automática de casos de testes, e essas três características são exploradas nessa pesquisa. Além disso, o uso da COFI minimiza a ambiguidade através dos modelos do comportamento dos serviços [Ambrosio 2007] e facilita o entendimento dos requisitos funcionais e operacionais.

Deste modo, a COFI foi utilizada parcialmente, pois para o método é relevante apenas o que está escrito na PUS e para obter essas informações basta utilizar os diagramas de estados de transições normais e os de exceções especificadas. Na confecção da TTER foi utilizada a união dos diagramas de estados normais e de exceções especificadas gerados pela aplicação da COFI em uma única tabela.

O método GeTTE, conforme pode ser observado na Figura 2, utiliza como uma de suas entradas (Entrada A) a TTER que é gerada a partir da aplicação parcial da COFI sobre os requisitos de um dos serviços da norma PUS.

Outra entrada (Entrada B, na Figura 2) para o método, uma Lista de Requisitos do Serviço (LRS), deve ser gerada a partir de uma análise do mesmo serviço da norma PUS utilizado na geração da TTER, porém nesta etapa a análise deve buscar por seus requisitos e transformá-los em declarações para a LRS. Um exemplo de declaração é “Em caso de sucesso na verificação somente é gerado um relato de telemetria se este for solicitado”. Em outras palavras, a LRS é um conjunto de declarações que deve conter todos os requisitos do serviço a ser modelado.

Na geração da LRS quando os requisitos possuem características opcionais, alternativas ou obrigatórias, estes devem ser modelados conforme essas características (Figura 1). Essa modelagem dos requisitos de acordo com suas características é baseada no conceito de características de Linha de Produto de Software (LPS) e deve ser feita para auxiliar na visualização das várias formas possíveis de se implementar o mesmo serviço. Cada uma das formas de se implementar o serviço deve gerar um conjunto de declarações para compor a LRS.

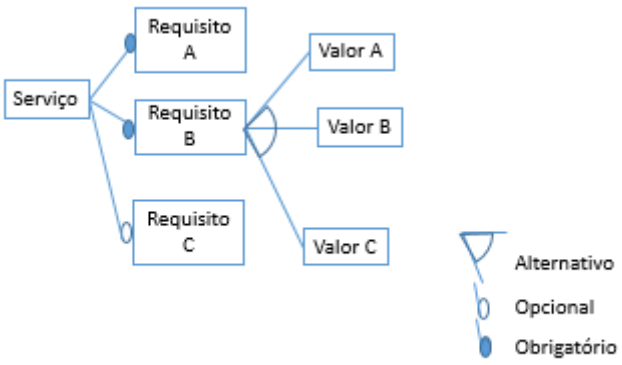

Figura 1. Modelo de Características Aplicável aos Requisitos conforme LPS.

Para verificar a completeza da LRS pode ser utilizada a as informações de descrição da tabela de entradas e saídas gerada ao se aplicar a COFI para a geração da TTER, pois as informações desta tabela devem estar representadas em forma de declarações nas LRS. Outra forma de verificar a completeza da LRS é através da análise dos casos de uso gerados quando se aplica a COFI para a TTER, pois os requisitos presentes na LRS devem cobrir todos os casos de uso gerados pela COFI. Porém, cabe ressaltar que essa atividade de geração das declarações ainda será mais elaborada.

O desenvolvedor do software a ser testado deverá selecionar um subconjunto da LRS que corresponde as declarações que representam as funções do software a ser testado. Este subconjunto é indicado como (Entrada C) na Figura 2.

O primeiro passo do método (Passo 1 ou Mapeamento de Referência da Figura 2) relaciona uma transição da TTER com todas as declarações da LRS relacionadas a ela, o que torna obrigatória a existência desta transição. Esse passo deve ser repetido para todas as transições da TTER.

No próximo passo (Passo 2 ou Mapeamento Final da Figura 2) o resultado do Passo 1 deve ser confrontado com o subconjunto da LRS (Entrada C) para gerar uma nova tabela de transição de estados derivada da TTER chamada Tabela de Transição de Estados Final (TTEF). A TTEF deve conter apenas as transições relacionadas com as declarações presentes na Entrada C que representam os requisitos do software a ser testado.

É importante notar que todos os passos da GeTTE são feitos para um único serviço por vez, ou seja, eles devem ser repetidos para todos os serviços da PUS.

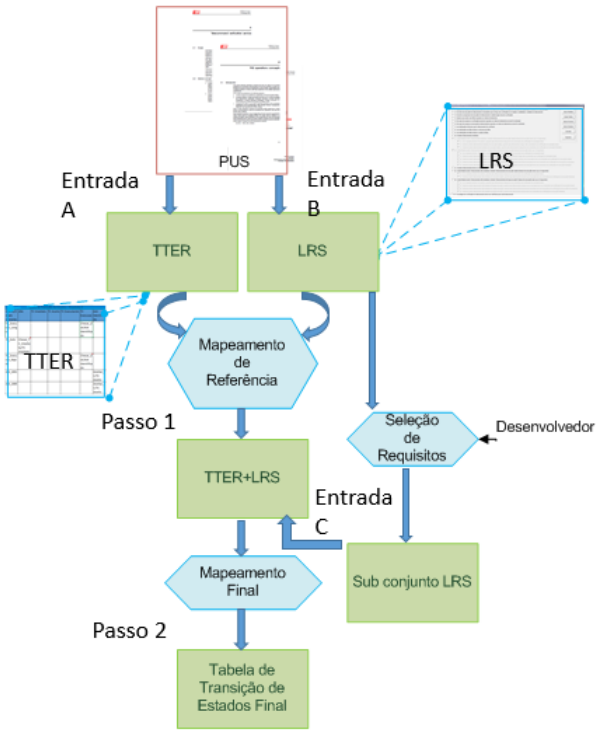

Figura 1. Método GeTTE.

3. Resultados e Discussão

Um dos resultados secundários da aplicação deste método é que o uso da LRS pode indicar se o serviço está presente, seja de forma integral ou parcial, no OBDH do nanosatélite a ser testado. Isso é possível, pois a LRS representa os requisitos do serviço e caso o desenvolvedor não selecione nenhuma das opções, significa que o serviço não foi implementado.

Outro resultado do método é a possibilidade de reuso das tabelas de transição de estados referentes ao serviço, seja parcial ou integral. Como consequência disso, tem-se reuso de modelos, o que indica uma redução de esforço de teste.

Finalmente, tem-se ainda que o uso de modelos minimiza a ambiguidade do texto formal através dos modelos do comportamento dos serviços e o uso desse método garante essa característica.

Esse método está em processo de desenvolvimento e está sujeito a mudanças conforme a necessidade. Neste momento estão sendo realizadas atividades de avaliação de resultados para poder validar esse método.

4. Conclusão

Esse método será utilizado como parte da abordagem de teste para V&V de software embarcado em nanosatélites que está sendo desenvolvida. Através dele é possível gerar máquinas de estados finitos referentes aos serviços da norma PUS que estão presentes no software a ser testado facilitando a geração dos casos de testes e, como consequência, facilitando a geração de testes de forma automática.

Em resumo, o método GeTTE gera uma tabela de transição de estados para as funções do OBDH de um nanosatélite que representem, de forma total ou parcial, um serviço da norma PUS.

Ao gerar a TTEF o método garante que o serviço a ser testado foi implementado (seja de forma parcial ou total) para o software a ser testado. Ele também minimiza a ambiguidade do texto formal por trabalhar com modelos de comportamento do serviço. E, também, ao se usar o método é garantido o reuso da TTER (seja de forma integral ou parcial) para vários software de nanosatélites.

A principal vantagem ao se utilizar esse método é a possibilidade de facilitar na geração de testes para nanosatélites, pois ele simplifica a tarefa de modelagem substituindo essa etapa por uma simples seleção de declarações que representem os requisitos implementados no software a ser testado. Isso é possível, pois a TTER e a LRS que representam o serviço completo serão geradas uma única vez e depois poderão ser usadas para quaisquer software de bordo de nanosatélite a ser testado. O que determinará a TTEF é a seleção das declarações a ser feita pelo testador com o apoio do desenvolvedor (aquele que conhece bem o software a ser testado).

Esse método ainda está em desenvolvimento e por isso ainda poderá ser alterado conforme surgir alguma necessidade. No momento, está sendo explorada uma forma de aperfeiçoar a geração da LRS para tornar essa lista o mais completa possível e de fácil entendimento para qualquer desenvolvedor de software de nanosatélite. Entre as formas de aperfeiçoamento

exploradas estão a validações da LRS com o apoio de desenvolvedores de projetos de software e especialistas em satélites.

Quando esse método estiver finalizado ele será aplicado para casos de estudos tais como NanosatC-BR2 e outros, visando validar sua usabilidade e avaliar a possibilidade de reuso dos modelos gerados em diferentes missões.

Agradecimentos: *Meus agradecimentos a CAPES, agência financiadora do meu doutorado.*

Referências

Ambrosio, A. M. (2005). "COFI: uma abordagem combinando teste de conformidade e injeção de falhas para validação de software em aplicações espaciais", 209 p. IBI: <6qtX3pFwXQZGivnJSY/HodCw>. (INPE-13264-TDI/1031). Tese (Doutorado em Computação Aplicada) - Instituto Nacional de Pesquisas Espaciais (INPE), São José dos Campos, 2005. Disponível em: <http://urlib.net/6qtX3pFwXQZGivnJSY/HodCw>.

Ambrosio, A. M. and Mattiello-Francisco, F. and Santiago Júnior, V. A. and Martins, E. (2007). "Designing fault injection experiments using state-based model to test a space software", In: LATIN-AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING (LADC), 3., Morelia, México. Proceedings... Berlin: Springer, 2007. p. 170-178. (INPE-15078-PRE/9987).

Ambrosio, A. M. and Martins, E. and Vijaykumar, N.L. and De Carvalho, S.V. (2006). "A Conformance Testing Process for Space Applications Software Services", Journal of Aerospace Computing, Information, and Communication (JACIC). American Institute of Aeronautics and Astronautics (AIAA). aceito em Janeiro de 2006. A ser publicado.

Ambrosio, A. M. and Mattiello-Francisco, F. and Santiago Júnior, V. A. and Martins, E. (2007). "Designing fault injection experiments using state-based model to test a space software", In: LATIN-AMERICAN SYMPOSIUM ON DEPENDABLE COMPUTING (LADC), 3., 2007, Morelia, México. Proceedings... Berlin: Springer, 2007. p. 170-178. (INPE-15078-PRE/9987).

Ambrosio, A. M.; Martins, E. (2006). "Space services: textual to formal description", In: INTERNATIONAL CONFERENCE ON SPACE OPERATIONS WILL BE HOSTED BY THE, 9TH., Rome, Italy. Proceedings... 2006. p. 11. CD-ROM. (INPE-14069-PRE/9238).

EUROPEAN COOPERATION FOR SPACE STANDARDIZATION (ECSS). (2016). "ECSS-E-ST-70-41C: Telemetry and telecommand packet utilization", 656p.

Greghi, J. G. and Martins E. and Carvalho, A. M. B. R. and Ambrosio, A. M.; Villani, E.. (2018). "From Natural Language Standard Documents to State Machines: Advantages and Drawbacks", Journal of Aerospace Information Systems, Vol. 15, No. 5, pp. 271-281.https://doi.org/10.2514/1.I010525

De Pádua Paula Filho, Wilson. (2003). "Engenharia de software", LTC.

Scholz, A. (2017). "CubeSat standards handbook". The LibreCube Initiative.