ontoELICERE: Um Conjunto de Ontologias para Representar os Elementos de uma Análise de Perigos em Sistemas Aeroespaciais

Glauco da Silva[1], Carlos Henrique Netto Lahoz[2]
[1]Instituto de Aeronáutica e Espaço
[2]Instituto Tecnológico de Aeronáutica
glaucogs@fab.mil.br, lahozchnl@fab.mil.br

Resumo: Este artigo apresenta o ontoELICERE, um conjunto de ontologias para representar os elementos de uma análise de perigos em sistemas aeroespaciais, suas relações, a técnica de análise utilizada, os perigos identificados e as medidas de mitigação sugeridas para cada perigo. Para se desenvolver o ontoELICERE, foi realizado um levantamento das informações necessárias para representar os elementos existentes na técnica ISTAR e quais informações são necessárias para a condução da análise de perigos. As ontologias foram criadas seguindo as recomendações do Método 101 e utilizou-se a linguagem OWL-DL. Com as ontologias é possível armazenar as medidas de mitigação selecionadas para que possam ser reutilizadas em outros projetos que venham a ser analisados, melhorando o processo de análise e o tempo de resposta. O ontoELICERE foi aplicado no acidente do Ariane 5 e foi útil na representação do modelo do sistema e na aplicação da análise de perigos. Todo conhecimento obtido com a análise é armazenado em bases de conhecimento para posterior consulta e reutilização em novas análises.

Palavras-chave: *Ontologias, Análise de perigos, ELICERE.*

1. Introdução

Atualmente, com o crescente número de aplicações críticas, como sistemas aeroespaciais, é necessário que a integridade do sistema como um todo seja garantida para que a missão não venha a falhar. A integridade pode ser relacionada a hardware, software, operação humana e ambiente. Neste sentido, a engenharia de segurança deve buscar o aprimoramento nos estudos e pesquisas de novas técnicas de análise de perigos, visto que poucas contribuições foram propostas tentando abranger esses quatro componentes e suas interações.

Nas investigações dos perigos são analisadas e descritas as potenciais situações em que o sistema pode ser levado a apresentar erros ou defeitos e consequentemente falhas. Estas análises buscam identificar comportamentos não planejados, troca de informações não previstas, entre outros problemas.

Várias técnicas são utilizadas para se realizar uma análise de perigos. Algumas técnicas bem conhecidas são FMEA (*Failure Mode and Effects Analysis*) [1], HAZOP (*Hazard and Operability Analysis*) [2] e FTA (*Fault Tree Analysis*) [3], que, apesar de serem consideradas tradicionais e conservadoras, ainda são muito utilizadas. Além destas técnicas, abordagens mais recentes, como STAMP (*Systems-Theoretic Accident Model and Processes*) [4], também tem sido adotadas. A maior dificuldade dessas abordagens é a dependência do conhecimento do analista, uma vez que é necessário que o condutor da análise conheça bem o sistema analisado.

Buscando minimizar estas dificuldades, em 2009 foi proposto o processo ELICERE [5]. Este processo integra a técnica de engenharia de requisitos orientados a metas ISTAR [6] com conceitos de técnicas de engenharia de segurança, como HAZOP e FMEA. O processo sugere como entrada um modelo do sistema a ser analisado e utiliza para análise um questionário que aborda características dos perigos potenciais de cada item do modelo. O resultado auxilia na identificação de metas de melhoria e medidas de mitigação, indicando atributos de qualidade ainda na fase de projeto.

O ELICERE foi proposto como um processo manual e gera um grande volume de informações. Além disso, os resultados das análises e o conhecimento adquirido no preenchimento dos questionários não podem ser utilizados para análises de outros projetos, tornando-se necessária a criação de uma base de dados para armazenar este conhecimento. Buscando suprir essas necessidades, foi criada a ferramenta de automatização do ELICERE, o PRO-ELICERE [7, 8, 9, 10]. Nesta ferramenta foram desenvolvidas duas bases de dados, a de atributos (ADB) e a de conhecimento (KB).

Visando tornar a ferramenta PRO-ELICERE mais robusta e inteligente, foi desenvolvido um conjunto de ontologias, chamado ontoELICERE, para auxiliar a condução da análise de perigos por meio de inferências e descoberta de conhecimento nas bases de dados que armazenam técnicas e medidas de mitigação. Estas ontologias são utilizadas para representar todo o processo de análise do PRO-ELICERE, desde o modelo do sistema até as medidas de mitigação sugeridas.

2. Metodologia

No desenvolvimento do ontoELICERE, foi realizado um levantamento das informações necessárias para representar os elementos existentes na técnica ISTAR e para a condução da análise de perigos. Para se criarem as ontologias foi adotada a linguagem OWL-DL [11], em conjunto com o método 101 [12].

A aplicação do ontoELICERE no acidente do Ariane 5 é apresentada na Seção 4.

3. ontoELICERE

O ontoELICERE é composto por três ontologias: ontoISTAR, utilizada para representar os elementos do sistema modelados com a técnica ISTAR e analisar seus relacionamentos em termos de perigos e dependências; ontoQUESTION, utilizada para analisar os desvios de intenção de projeto sobre os campos do questionário; e ontoMITIGATION, aplicada para auxiliar na criação de medidas de mitigação. As ontologias foram criadas com o software livre Protégé[1] e são compostas por classes, atributos e relacionamentos. A Figura 1 apresenta as classes do ontoELICERE.

1 http://protege.stanford.edu/

DOI: 10.22478/ufpb.2595-0622.2018v2n2.40963

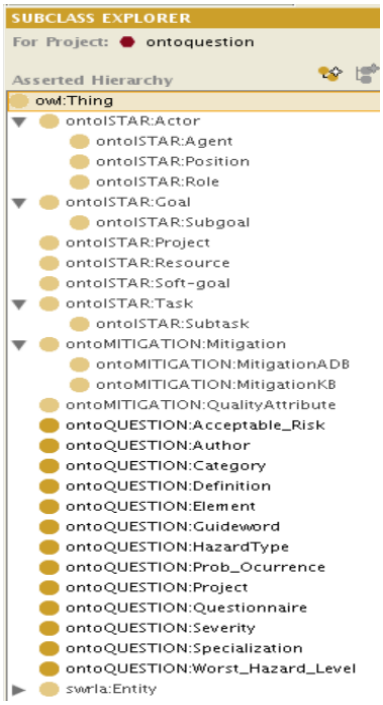

Figura 1: Classes do ontoELICERE.

3.1. ontoISTAR

A ontoISTAR foi criada baseada nos elementos existentes na técnica ISTAR [6], visando representar os elementos de um modelo ISTAR em instâncias da ontologia para que a análise de perigos seja conduzida.

Todas as classes definidas nesta ontologia são baseadas nos elementos que podem existir em um modelo ISTAR: ator, recurso, meta, meta-soft e outros. A ontologia também representa os relacionamentos entre os elementos, como níveis de dependência entre os elementos e decomposição de tarefas. As classes são utilizadas para conduzir a análise de perigo, permitindo definir a severidade, a palavra guia e assim por diante.

Uma das principais vantagens de se fazer uso de ontologias é a capacidade de inferir conhecimento para melhorar a condução da análise, incluindo, por exemplo, a possibilidade de detecção dos elementos que possuem maior número de dependências. Buscando identificar essa característica, foram definidas assertivas que apontam a quantidade de dependências de cada elemento. Exemplos dessas assertivas, representadas por descritores lógicos, são apresentados a seguir.

```
(ontoISTAR:Resource(?x) ∧ ontoISTAR:dependencyF
rom(?x,?y))→(sqwrl:select(?x)∧ sqwrl:count(?y))
(ontoISTAR:Resource(?x) ∧ ontoISTAR:dependencyT
o(?x,?y))→(sqwrl:select(?x) ∧ sqwrl:count(?y))
```

Essas regras realizam a contagem de dependências para um elemento, por exemplo, se "*?x*" é um recurso (*ontoISTAR:Resource(?x)*) e ele possui dependência de um ou mais elementos (*ontoISTAR:dependencyFrom(?x,?y)*), ocorre a contagem deste número (*query:count(?y)*), na qual "*?y*" é o número de dependências de "*?x*".

3.2. ontoQUESTION

As classes da ontoQUESTION foram criadas para representar os questionários e seus campos que devem ser preenchidos durante a análise de perigos. Além de permitir a condução da análise, a ontoQUESTION também permite a descoberta de novas informações a respeito dos perigos analisados, como por exemplo, mecanismos para classificar o risco associado aos perigos analisados. Todos os riscos são associados com a possibilidade de ocorrência (*ontoQUESTION:Hazard_Probability*) e um nível de severidade (*ontoQUESTION:Hazard_Severity*). A seguir são apresentados alguns exemplos de como essa classificação pode ser utilizada.

Um perigo com ocorrência "*Improbable*" e severidade "*Catastrophic*" classifica o perigo com o risco "*Acceptable*". Um perigo com ocorrência "*Frequent*" e severidade "*Major*" classifica o perigo com o risco "*Intolerable*".

Com base nessas características, algumas assertivas foram definidas na ontologia para classificar o risco do perigo analisado. Essas assertivas, são representadas na ontologia por descritores lógicos, conforme apresentado no exemplo a seguir.

```
(ontoQUESTION:Element(?x) ∧ ontoQUESTION:Hazard
_Probability(?x,ontoQUESTION:Probability_Freque
nt) ∧ ontoQUESTION:Hazard_Severity(?x,ontoQUEST
ION:Severity_Major))→(ontoQUESTION:Classificatio
n_of_Risk(?x,ontoQUESTION:Risk_Intolerable))
```

Esta regra define que se "*?x*" é um elemento analisado (*ontoQUESTION:Element*) e possui a probabilidade de ocorrência (*ontoQUESTION:Hazard Probability()*) "*ontoQUESTION:Probability_Frequent*" e severidade (*ontoQUESTION:Hazard_Severity()*) "*ontoQUESTION:Severity_Major*", então "*?x*" possui a classificação de risco (*ontoQUESTION:Classification_of_Risk()*) "*ontoQUESTION:Risk_Intolerable*". Esta regra permite determinar o risco de cada perigo indicado na análise e então priorizar as análises dos perigos para aqueles com risco "*Intolerable*".

Assim, quando um perigo apresentar um risco "*Intolerable*", significa que o questionário associado a ele deve ser preenchido pelo analista/engenheiro, uma vez que este perigo pode causar um comprometimento do sistema, sendo assim é necessário aplicar alguma medida de mitigação para reduzir ou eliminar o risco do sistema deixar de funcionar.

3.3. ontoMITIGATION

Para se identificarem as medidas de mitigação, o sistema pode utilizar as bases de dados de atributos (ADB) e a base de conhecimentos (KB). A ADB foi criada utilizando-se trabalhos da literatura [13, 14, 15, 16], que conduziram levantamentos de perigos, suas causas e suas possíveis mitigações. A KB foi construída por meio de medidas de mitigação utilizadas em projetos já analisados, permitindo o reuso de mitigações já adotadas. Todas as medidas de mitigação existentes são representadas por instâncias desta ontologia.

Para popular a KB por meio da ontologia, foi criada uma assertiva que define que qualquer medida de mitigação adotada para um questionário deve ser adicionada na KB, incluindo as informações do projeto em que ela foi aplicada e para qual perigo foi selecionada. Essa assertiva é representada na ontologia por descritores lógicos, conforme apresentado a seguir.

DOI: 10.22478/ufpb.2595-0622.2018v2n2.40963

```
(ontoQUESTION:Questionnaire(?x) ∧ ontoQUESTIO:q
uestionnaireMitigation(?x,?y) ∧ ontoQUESTION:Or
iginProject(?x,?z)) → (ontoMITIGATION:Mitigatio
nKB(?y) ∧ ontoMITIGATION:Project(?y,?z))
```

Essa assertiva indica que se existe um questionário "*?x*" (*ontoQUESTION:Questionnaire(?x)*) e "*?y*" é uma medida de mitigação adotada para "*?x*" (*ontoQUESTION:questionnaireMitigation(?x,?y)*), e "*?z*" é o projeto original do questionário "*?x*" (*ontoQUESTION:OriginProject(?x,?z)*), então "*?y*" é uma medida de mitigação que deve ser inserida na base de conhecimento (*ontoMITIGATION:MitigationKB(?y)*) e a informação de qual projeto utilizou aquela mitigação deve ser inserida acompanhada de a medida adotada (*ontoMITIGATION:Project(?y,?z)*).

Quando o sistema busca por uma medida de mitigação para um perigo, primeiramente é realizada uma busca na KB, visando fazer o reuso de uma mitigação já adotada para um perigo similar de outro projeto. Caso o sistema não encontre a mitigação na KB, então uma busca é realizada na ADB.

4. Uma aplicação do ontoELICERE

A família de veículos lançadores de satélites Ariane do *Centre National D'etudes Spatiales* (CNES), tem como objetivo colocar satélites artificiais em órbitas geoestacionárias, e também enviar cargas para órbitas de baixa altitude.

Em 4 de junho de 1996, o primeiro voo do lançador Ariane 5 não foi bem sucedido. Aproximadamente 40 segundos após o início da sequência de voo, a uma altitude de cerca de 2.7 Km, o lançador se desviou de sua trajetória, se quebrou e explodiu [18]. O relatório de acidente [17] descreve a "causa primária" como a perda total das informações de orientação e atitude 37 segundos após o início da sequência de ignição do motor principal (30 segundos após a decolagem). A perda de informação ocorreu devido a erros de especificação e de projeto no software do sistema de referência inercial [18]. O software foi reutilizado a partir do Ariane 4 e incluiu funções que não eram necessárias para o Ariane 5. As perdas do acidente incluíram cinco bilhões de dólares de carga útil [17].

Após estudos sobre a estrutura do Ariane 5 e a identificação de seus elementos, foi desenvolvido o modelo ISTAR para o veículo lançador e identificadas as metas, metas-soft, recursos e tarefas, assim como as dependências existentes entre eles.

A partir do modelo, a ontoISTAR foi utilizada para representar os elementos existentes, tornando possível a condução da análise de perigos. As instâncias representam os elementos e suas características, bem como suas dependências com outros itens do modelo.

Após a representação do modelo pela ontoISTAR, a ontoQUESTION foi utilizada para representar a Tarefa "*Calculate inertial data*", que foi identificada como um dos elementos mais críticos do sistema. Vale ressaltar que a ontologia representa todos os elementos do sistema, uma vez que outros perigos podem existir.

Após o preenchimento do questionário, são apresentadas as sugestões de mitigação, representadas por instâncias da ontoMITIGATION. Primeiro as da KB, e caso não exista nenhuma ou a apresentada não seja satisfatória, são apresentadas as da ADB. O objetivo é identificar se na ADB existem técnicas que possam produzir recomendações adequadas ao problema. Após a seleção de uma ou mais medidas, os resultados são inseridos na KB como mitigações já utilizadas, assim é possível que projetos futuros possam fazer o reuso desta informação.

As recomendações indicadas no relatório de investigação [17] também foram inseridas na KB, pois são conhecimentos já reportados e portanto podem ser utilizados em projetos futuros semelhantes.

Para o caso analisado, foi verificado que o cálculo incorreto ocorreu devido ao reuso de uma parte do software do Ariane 4. Para este caso não foi encontrada nenhuma medida de mitigação na KB, portanto a busca foi realizada na ADB, e teve como sugestão a medida de mitigação "*Reuse checklist*" (Figura 2), indicando recomendações que devem ser seguidas para garantir o reuso de código de maneira eficiente. Alguns trabalhos abordam o reuso de software [19, 20], mas basicamente as recomendações envolvem testes intensivos nos códigos reutilizados e risco aceitável desses códigos

ontoMITIGATION:Mitigation_Task_Software_...	
ontoMITIGATION:Project =	ontoQUESTION:Project_Ariane_5
ontoMITIGATION:QualityAttributeADB =	ontoMITIGATION:QualityAttribute_Reusabil...
ontoMITIGATION:MitigationHazard =	ontoQUESTION:Hazard_Task_Incorrect
ontoMITIGATION:MitigationHazardType =	ontoQUESTION:HazardType_Software
ontoQUESTION:MitigationDescription =	Reuse Checklist

Figura 2: Medida de mitigação "Reuse checklist" representada pelo ontoMITIGATION.

5. Conclusões

A necessidade de se realizarem análises de perigos em sistemas críticos tem motivado o desenvolvimento de novas ferramentas para auxiliar nesta tarefa. Os desafios de pesquisa englobam técnicas, geralmente utilizadas em dispositivos mecânicos ou eletrônicos, que precisam ser adaptadas para software.

Além deste desafio, também é necessária a criação de ferramentas que auxiliem na condução da análise. Devido a essa necessidade, pensou-se na criação do PRO-ELICERE, que busca automatizar o processo ELICERE para análise de perigos.

A versão inicial do PRO-ELICERE foi utilizada para introduzir a aplicação do ontoELICERE, mostrando como ele deve operar com esta nova funcionalidade. O preenchimento do questionário de forma eletrônica, com valores pré-fixados, não gera ambiguidade nem dúvidas quanto aos elementos que devem ser inseridos.

O processo se mostrou robusto para realizar as recomendações de mitigação, com base nas técnicas de mitigação existentes na ADB e KB. Isto mostra que o uso das ontologias é capaz de auxiliar na condução da análise. A utilização do ontoELICERE se apresenta como uma ferramenta poderosa para suporte a decisão, uma vez que permite a adoção ou não da medida sugerida para o perigo.

Além disso, o ontoELICERE se mostrou promissor na representação dos elementos existentes no sistema analisado. As ontologias são úteis para a realização de inferências, permitindo que novas informações e novos conhecimentos sobre o sistema analisado sejam obtidas.

Bibliografia

[1] Stamatis, D. H. (2003) Failure mode and effect analysis: FMEA from theory to execution. ASQ Quality Press.

[2] Redmill, F. et al. (1999) System safety: HAZOP and software HAZOP. Wiley Chichester.

[3] Lee, W.-S. et al. (1985). Fault tree analysis, methods, and applications: A review. *IEEE transactions on reliability* 34(3):194-203. DOI: 10.1109/TR.1985.5222114

[4] Leveson, N. (2004) A new accident model for engineering safer systems. *Safety science*, 42(4):237-270. DOI: 10.1016/S0925-7535(03)00047-X

[5] Lahoz, C. H. N. (2009) ELICERE: O processo de elicitação de metas de dependabilidade para sistemas computacionais críticos: Estudo de caso aplicado à área espacial. Tese de Doutorado. Universidade de São Paulo. Programa de Pós-Graduação em Engenharia Elétrica.

[6] Yu, E. S. (1995) Modeling Strategic Relationship for Process Reengineering. Tese de Doutorado. Toronto University.

[7] Silva, G.; Lahoz, C. H. N. (2013) A new process for space computer system dependability analysis. In *64th International Astronautical Congress*, Beijing, China. IAF.

[8] Silva, G.; Lahoz, C. H. N. (2013) Pro-elicere: A study for create a new process for dependability analysis of space computer systems. In *Sixth IAASS Conference - Safety is Not an Option*, Montreal, Canada. ESA Communications.

[9] Pivetta, T. A. (2016) Uma estrutura de base de dados para o processo de análise de perigos - pro-elicere. Dissertação de Mestrado. Instituto Tecnológico de Aeronáutica. Programa de Pós-Graduação em Ciências e Tecnologias Espaciais.

[10] Pivetta, T. A. et al. (2016) Pro-elicere: A hazard analysis automation process applied to space systems. *Journal of Aerospace Technology and Management* 8(3):328-338. DOI: 10.5028/jatm.v8i3.609

[11] Horrocks, I. et al. (2003) From shiq and rdf to owl: the making of a web ontology language. *Web Semantics: Science, Services and Agents on the World Wide Web* 1(1):7-26. DOI: 10.1016/j.websem.2003.07.001

[12] Noy, N. F.; McGuinness, D. L. (2001) Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Knowledge Systems, AI Laboratory, Stanford University, Stanford, EUA.

[13] Avizienis, A. et al. (2004) Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing* 1(1):11-33. DOI: 10.1109/TDSC.2004.2

[14] Bass, L. (2007). Software architecture in practice. Pearson Education India.

[15] Romani, M. A. S. et al. (2010). Identifying dependability requirements for space software systems. *Journal of Aerospace Technology and Management* 2(3):287-300. DOI: 10.5028/jatm.2010.02037810

[16] Ericson, C. A. et al. (2015) Hazard analysis techniques for system safety. John Wiley & Sons.

[17] Lions, J.-L. et al. (1996) Ariane 5 flight 501 failure. Technical report, European Space Agency.

[18] Leveson, N. G. (2004) Role of software in spacecraft accidents. *Journal of Spacecraft and Rockets* 41(4):564-575. DOI: 10.2514/1.11950

[19] Leach, R. J. (1997) Software Reuse: methods, models, and costs. McGraw-Hill New York.

[20] Rothenberger, M. A. et al. (2003) Strategies for software reuse: A principal component analysis of reuse practices. *IEEE Transactions on Software Engineering* 29(9):825-837. DOI: 10.1109/TSE.2003.1232287