Uma Proposta de Sistemática para a Análise de Segurança de software Crítico Embarcado Aeroespacial

Elio Lovisi Filho
APPI Informática Ltda. Rio de Janeiro - RJ
(021) 540 9100 elio@appi.com.b

Adilson Marques da Cunha
Divisão de Ciência da Computação Instituto Tecnológico de Aeronáutica São José dos Campos - SP
(012) 347 5896 cunha@comp.ita.cta.br

Abstract

In spite of the increasing integration of Computers into Aerospace Systems, there is no evidence of existing effective Systematic for realize the Analysis of Aerospace Embedded Critical software, able to aid in obtain appropriate levels of software Safety. The lack of this kind of Systematic besides increasing software complexity, cost and development time has caused considerable waste of resources. This article presents the main ideas being investigated to compose a new Systematic for Analysis, that attends software Safety requirements. The authors believe that acceptable levels for the Safety of Aerospace Embedded Critical software can be only obtained if new solutions are found for software Safety problems. The Systematic for Analysis proposed in this article intends to represent an added value effort into this direction.

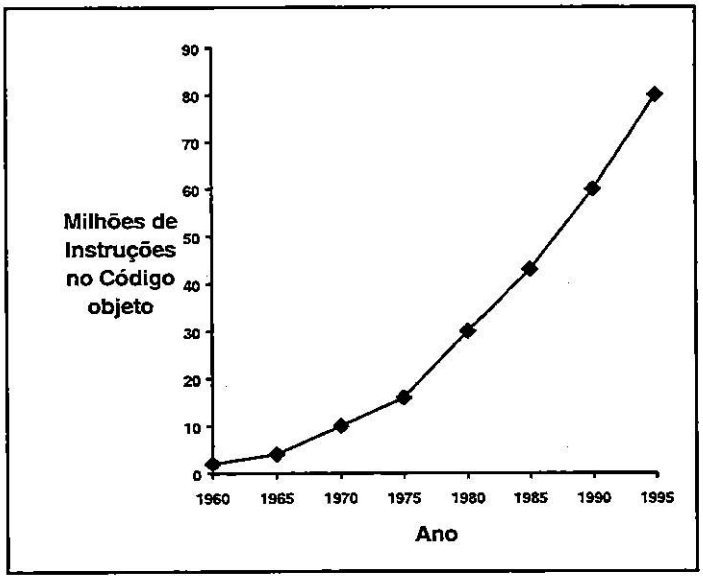
1 Introdução

Atualmente, observa-se uma crescente incorporação dos computadores aos Sistemas Aeroespaciais, utilizado-se o mesmo para a realização das mais diversas funções. Uma importante aplicação do computador nesses sistemas é o apoio aos processos de controle e tomada de decisão, proporcionando consideráveis ganhos tanto no desempenho quanto na eficácia dos Sistemas Aeroespaciais.

O emprego dos computadores para auxílio ao controle de Sistemas Aeroespaciais exige uma atenção especial em relação à Segurança, uma vez que estes sistemas são considerados como Críticos, pois uma falha no seu funcionamento pode levar a grandes perdas, tanto de ordem econômica, como ambiental ou humana.

A Figura 1, a seguir, apresenta o crescimento do uso de software Embarcado em um importante tipo de Sistema Crítico.

Figura 1: Utilização de Software Embarcado no Programa Espacial Norte-Americano [Moura e Santellano, 1999]



Pode-se observar, na Figura 1, um crescimento exponencial do emprego de software no programa Aeroespacial norte-americano.
A crescente utilização dos softwares Críticos Embarcados nos Sistemas Aeroespaciais deve-se, principalmente, ao desenvolvimento da tecnologia digital, ao controle computadorizado de tarefas, à reusabilidade do software, à miniaturização, à redução de custos e à facilidade de integração dos Sistemas computadorizados.

O emprego de computadores para apoio aos processos de controle e decisão de Sistemas Aeroespaciais salienta a necessidade de se desenvolver ou adaptar novas tecnologias, compostas de métodos, técnicas, ferramentas e métricas, para garantir a Segurança dos mesmos. Busca-se assim evitar que uma falha do computador venha a comprometer o funcionamento ou desempenho de todo o sistema.
Considera-se o software como o componente de um computador mais passível à ocorrência de falhas, exigindo a realização de seu desenvolvimento com Garantia de Segurança, ou seja, garantindo que o mesmo irá funcionar em um sistema sem resultar em riscos inaceitáveis. Esse tipo de estudo faz parte da área de Segurança de software (software Safety).
Neste artigo propõe-se a definição de uma Sistemática para a realização de uma Análise de Segurança de um software Crítico Embarcado Aeroespacial (SCEA), determinando-se as restrições de Segurança de software para este tipo de aplicação.

2. Segurança de Software

De acordo com o que foi apresentado anteriormente, a Segurança de software envolve estudos para garantir que o software execute suas funções sem acarretar risco inaceitável para a Segurança do sistema, dos usuários e do meio ambiente em que o sistema está inserido.
Portanto, pode-se considerar a Segurança de software como um fator explícito de Qualidade, uma vez que, caso ela não se verifique para um software, ele não atenderá às expectativas do usuário [Lovisi e Cunha, 1999].
A Garantia de Segurança de software compõe-se de uma seqüência de atividades planejadas que visam assegurar a imunidade da aplicação de software em relação à possibilidade de ocasionar acidentes que comprometam a vida humana, o meio ambiente, ou a propriedade [Moura, 1996].
Define-se Estratégia de Segurança de software como um conjunto de atividades que visam garantir que a Segurança de software é devidamente tratada no desenvolvimento ou manutenção do mesmo. O padrão MIL-STD-498 recomenda a definição de uma Sistemática de Segurança como estratégia de Segurança de software [Santellano et al, 1998].
Neste artigo será apresentada uma Sistemática para Análise de Segurança, que se aplica somente durante o desenvolvimento do software Crítico. Dessa forma, procura-se atestar níveis mais adequados de Segurança ao processo de desenvolvimento de software, desde as primeiras fases de sua concepção.
Para o desenvolvimento deste estudo, é necessário apresentar algumas definições relacionadas à Segurança de software [Pôrto e De Bortoli, 1997; Ippolito e Wallace, 1995]:

- Acidente (Mishap) é um evento não planejado que causa perdas humanas, danos ao meio ambiente ou danos aos equipamentos;
- Defeito (Fault) é uma imperfeição existente no código fonte do programa que, ao ser ativada, pode produzir erro;
- Erro (Error) é a manifestação física de um defeito que pode gerar falha;
- Falha (Failure) é a inabilidade do software em cumprir algum requisito operacional de sua responsabilidade, ou ainda, a produção de um efeito indesejado do software;
- Estados Inseguros ou Perigos (Hazards) são os estados do sistema que, combinados a certas condições externas, podem levar a acidentes;
- Um Sistema Crítico (Critical System) apresenta restrições relacionadas a um determinado fator, como por exemplo: financeiro, tempo, capacidade do equipamento e segurança. Abordam-se aqui apenas os Sistemas Críticos quanto à Segurança, ou seja, no contexto deste artigo qualquer referência a Sistemas Críticos relaciona-se somente a Sistemas Críticos quanto à Segurança; e
- Risco (Risk) para esta área do conhecimento é definido em função dos seguintes fatores:
  - da probabilidade de ocorrência de Estados Inseguros;
  - da probabilidade dos Estados Inseguros acarretarem acidentes; e
  - pior perda possível associada a esse acidente.

3 Desenvolvimento de Software

Define-se software como: uma seqüência de instruções que, quando executadas, produzem a função e o desempenho desejados; as estruturas de dados necessárias para a realização destas instruções; e toda a documentação que descreve a operação e o desenvolvimento do mesmo [Pressman, 1995].

O ciclo de desenvolvimento de um software compreende todas as fases, que vão desde a concepção até seus ensaios ou testes.
Os autores propõem a inclusão de um procedimento para a Análise de Segurança de software ao ciclo de

desenvolvimento, visando tratar o problema da Segurança de software desde o início de sua implementação.
Nessa fase busca-se determinar e avaliar as falhas do software que possam levar o Sistema, que ele faz parte, a um Estado Inseguro. A Análise de Segurança de software possui grande importância, principalmente para softwares componentes de Sistemas Críticos. Essa análise é desenvolvida a partir das informações obtidas na Análise de Sistemas.
Enquanto a fase de Análise de Sistemas de uma metodologia visa a identificação das funções que o software deve executar, a Análise de Segurança concentra-se em que o software não deve executar [Leveson, 1991].
Para realizar esta análise com sucesso necessita-se também de uma sistemática bem definida, composta de métodos e técnicas, objetivando a identificação e a avaliação dos Estados Inseguros.

No próximo item, apresenta-se as atividades que constituem a Análise de Segurança de software, de acordo com os estudos publicados nas referências [Leveson, 1991; Leveson e Harvey, 1983; Moura, 1996].

4 Análise de Segurança de Software

Como apresentado anteriormente, a Análise de Segurança de software baseia-se em determinar quais situações podem levar o Sistema a um Estado Inseguro. Esta análise possui diversas atividades, visando: identificar os Estados Inseguros e as falhas que os originaram; determinar o Fator Crítico (Criticality) dos mesmos; e ainda, avaliar a aceitabilidade dos níveis de Segurança do Sistema.
A Figura 2 a seguir apresenta todas as atividades componentes da Análise de Segurança de software, bem como as informações de entrada e de saída de cada uma destas atividades

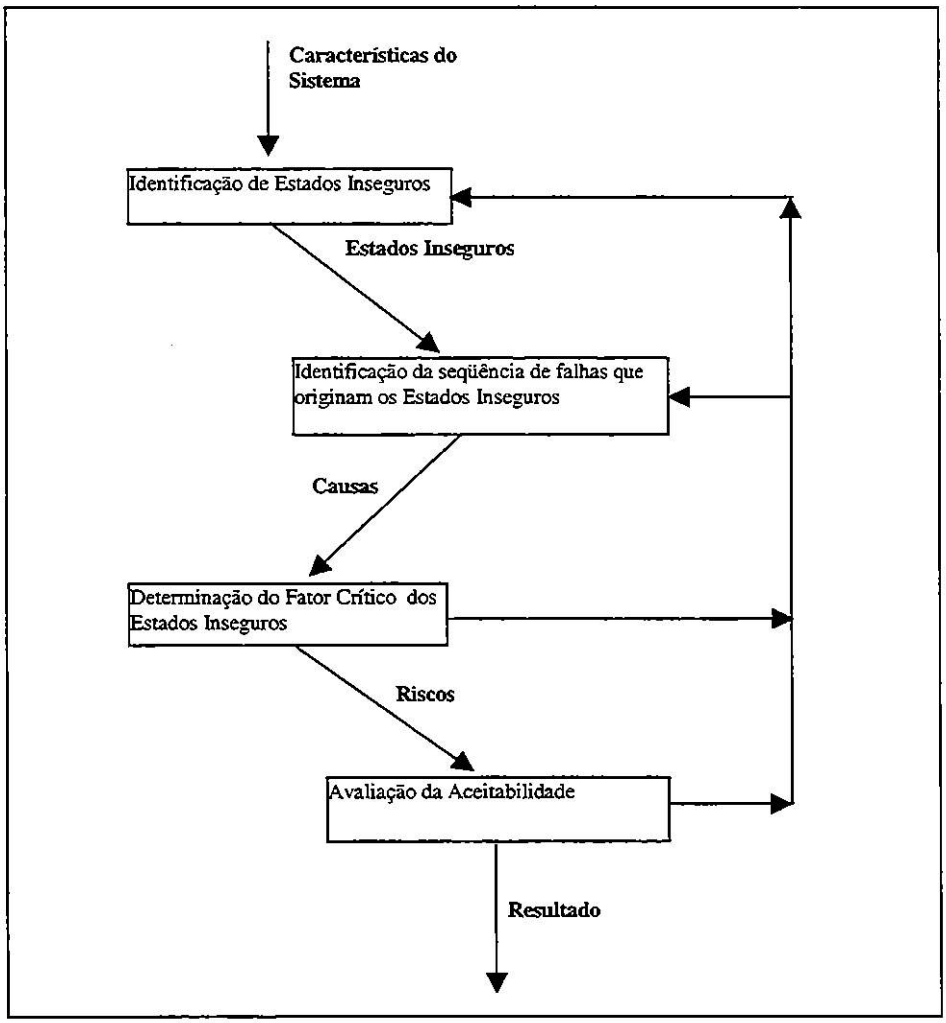


Figura 2: Procedimento para Análise de Segurança de Software

As atividades da Análise de Segurança de software devem se repetir à medida que a quantidade de informações aumenta; logo, não se pode esperar que as mesmas, obedeçam a uma rígida ordem cronológica, devendo-se revisar estas análises a cada modificação no projeto do sistema.
No início do desenvolvimento de um software Crítico Embarcado, elabora-se a Lista Preliminar de Inseguranças, contendo as possíveis falhas para este tipo de software. Esta lista é desenvolvida a partir de informações existentes sobre Análises de Segurança de software similares.
À medida que o desenvolvimento avança, realizam-se outras análises também para a identificação dos Estados Inseguros e de suas falhas causadoras:

- A Análise Preliminar de Insegurança (Preliminary Hazards Analisysis), que visa determinar os subsistemas críticos e se possível propor alternativas de controle. Baseia-se nos dados existentes e na experiência dos analistas para desenvolver uma análise em alto nível das principais funções e interfaces;
- A Análise de Insegurança de Subsistemas (Subsystem Hazards Analisysis) objetiva a identificação dos Estados Inseguros e suas falhas no projeto de cada subsistema e de sua interface. Concentra-se basicamente em aspectos tais como: desempenho, degradação no funcionamento e falhas funcionais, procurando determinar os modos de falhas e seus efeitos;
- A Análise de Insegurança de Sistema (System Hazards Analysis) identifica os Estados Inseguros e suas falhas, associados às interfaces entre os subsistemas, incluindo erros potenciais humanos; e
- A Análise de Insegurança na Operação e Suporte (Operation and Support Hazard Analysis) avalia os Estados Inseguros ocorridos durante as etapas de uso e manutenção do sistema, especialmente àqueles provenientes da interação do homem com o Sistema.

Para realizar estas análises, é necessária a utilização de algumas técnicas para identificação de Estados Inseguros, tais como: Hazop; SFTA; FMECA e Redes Petri Temporais [Ippolito e Wallace, 1995].
Como as técnicas para Análise de Segurança de software são adaptadas de outras áreas da tecnologia, geralmente é necessária a utilização conjugada das mesmas, para a identificação apropriada dos Estados Inseguros.
Deve-se também conhecer as características peculiares do tipo de aplicação em desenvolvimento, para determinar as técnicas mais adequadas a serem utilizadas nas análises de segurança, observando-se também, as potencialidades e restrições das mesmas.

Cabe à Sistemática para Análise de Segurança, determinar as técnicas a serem aplicadas e a seqüência de utilização das mesmas, de acordo com as características da aplicação.
Após a identificação dos Estados Inseguros, realiza-se uma avaliação dos mesmos quanto ao seu Fator Crítico, considerando fatores como Severidade do acidente ativado pelo Estado Inseguro e sua Probabilidade de ocorrência [Leveson, 1991].
A determinação do Fator Crítico dos Estados Inseguros consiste em classificar os mesmos, de acordo com a Severidade (Severity) do acidente causado por eles (Catastrófico, Crítico, Marginal e Menor) e a Probabilidade (Probability) de ocorrência deste acidente (Freqüente, Provável, Ocasional, Remota e Improvável).
A Tabela 1 mostra a classificação dos Estados Inseguros quanto ao Fator Crítico proposta no padrão britânico Int Def Stan 00-56 [Ministry of Defense, 1997].

Fator Crítico				
	Tipo de Severidade			
Faixa de Probabilidade	Catastró-fica	Crítica	Marginal	Desprezí-vel
Freqüente	T4	T4	T3	T2
Provável	T4	T3	T3	T2
Ocasional	T3	T3	T2	T2
Remota	T3	T2	T2	T1
Improvável	T2	T2	T1	T1

Tabela 2 - Classificação do Acidente quanto ao Fator Crítico [Ministry of Defense, 1997].

Cada nível de Fator Crítico exige um tratamento diferenciado para garantir a Segurança do software. Por exemplo, um estado do nível T4 necessita de atenção especial exigindo a utilização de estruturas de programação para evitar, controlar ou recuperar sua ocorrência, ao passo que, um estado do nível T1 pode não demandar tantos cuidados.
Deve-se observar também que, devido à falta de dados numéricos para o software, geralmente desconsidera-se a sua classificação quanto à probabilidade.
Após o término da determinação do Fator Crítico, avaliam-se os Estados Inseguros não evitados, controlados ou recuperados completamente, para determinar se os mesmos realmente não representam uma ameaça à Segurança de software, garantindo assim, que o mesmo apresenta um nível aceitável de risco.

5 Software Crítico Embarcado Aeroespacial - SCEA

As considerações realizadas até então, não se relacionam a nenhum tipo peculiar de software. A partir deste ponto,

aborda-se neste artigo, apenas os softwares Críticos Embarcados Aeroespaciais (SCEA's).
Denomina-se software Crítico, aquele utilizado em computadores componentes de Sistemas Críticos. Este tipo de software exige um desenvolvimento cuidadoso para garantir a Segurança de software necessária, pois, uma falha na sua execução pode originar um acidente com grandes perdas humanas, econômicas e ambientais.
Um software Crítico Embarcado Aeroespacial (SCEA) é empregado em computadores Embarcados em Sistemas Aeroespaciais. É importante ressaltar que essa classe engloba, tanto o software componente de computadores Embarcados nas Aeronaves, quanto àqueles utilizados nos equipamentos de solo [Moura e Santellano, 1999].
Essa aplicação particular do software Crítico Embarcado, exerce as mais diversas funções em Sistemas Aeroespaciais, principalmente, o apoio à tomada de decisões e controle de outros componentes.
Deve-se aqui ressaltar a diferença entre um software para aplicação civil, onde é necessário evitar a ocorrência de acidentes que possam causar perdas de vidas, de um software para aplicação militar, onde é necessário garantir que o Sistema irá completar sua missão com êxito.

5.1 As Características do Software Crítico Embarcado Aeroespacial - SCEA

O software Crítico Embarcado Aeroespacial (SCEA), atualmente, vem sendo empregado em novas funções para o software, em substituição a componentes eletrônicos que apresentam alta Confiabilidade e Segurança.
Como os Sistemas Aeroespaciais possuem alto Fator Crítico, os seus componentes de software possuem algumas características peculiares, de acordo com [Parnas, 1986; Moura et al, 1996], que devem ser observadas para o seu desenvolvimento.
Esse tipo de software possui algumas restrições em relação ao tamanho do código, à capacidade da memória e ao uso do processador. Ele deve também, implementar técnicas que permitam a recuperação de dados perdidos ou danificados por interferência eletromagnética no hardware.
A utilização do software Crítico Embarcado em Sistemas Aeroespaciais, geralmente, objetiva: a execução de complexos cálculos matemáticos, muitos dos quais não podem ser realizados por seres humanos em tempo hábil; a realização de operações de auxílio e monitoramento do controle de todo o Sistema; a entrada de dados no Sistema; e a implementação do controle das funções básicas do Sistema.
O software deve ter alta qualidade, confiabilidade, segurança e tolerância a falhas, pois qualquer erro em sua execução pode levar a um acidente com grandes perdas.

5.2 O Desenvolvimento do software Crítico Embarcado Aeroespacial - SCEA

Devido às características do software Crítico Embarcado Aeroespacial, algumas recomendações principais devem ser observadas para o seu desenvolvimento, a saber: a utilização de um enfoque metodológico para o desenvolvimento do software; a adoção e adaptação de padrões que prevêem considerações sobre segurança; o uso de ferramentas CASE para apoio ao desenvolvimento; e uma documentação clara e abrangente de todo projeto [Parnas, 1985; Parnas, 1986; Leveson e Harvey, 1983; Moura et al, 1996].
Além disso, a adoção de procedimentos de testes sistemáticos com uso de simuladores de software e de estratégias de certificação, uma avaliação quantitativa do software e a utilização de métodos formais podem ser necessários, de acordo com o grau de Confiabilidade e de Segurança exigidos do software.
Um estudo das interdependências entre software e hardware, levando em consideração as características da aplicação, e também uma investigação dos acidentes envolvendo softwares semelhantes podem ser importantes fontes de informação, permitindo a identificação das fraquezas e prioridades do processo de desenvolvimento.
A abordagem desse trabalho de pesquisa concentra-se exclusivamente, na Análise de um SCEA, visando observar a Garantia de sua Segurança. Para isso, deve-se utilizar uma Sistemática para Análise que propicie a identificação e a avaliação dos Estados Inseguros, auxiliando assim na garantia de níveis apropriados de Segurança à aplicação.

6 Sistemática para Análise de Segurança de Software Crítico Embarcado Aeroespacial

A partir das características do SCEA apresentadas no item anterior, desenvolveu-se a Sistemática para realização da Análise de Segurança deste tipo de software. Esta sistemática determina as técnicas mais adequadas para a realização da análise e a seqüência de utilização das mesmas.
Como citada anteriormente, a Análise de Segurança de software inicia-se com a preparação de uma Lista Preliminar de Inseguranças. Esta lista também pode ser elaborada na fase de Análise de Requisitos, integrando as restrições de Segurança aos requisitos do Sistema.
A partir da Lista Preliminar de Inseguranças, realiza-se a Análise Preliminar de Insegurança, visando distinguir os Subsistemas Críticos. O produto dessa análise é uma lista, contendo os componentes críticos do Sistema e seus possíveis Estados Inseguros.
A Análise de Insegurança de Subsistemas busca identificar novos Estados Inseguros nos Subsistemas Críticos, utilizando para isso as técnicas: Análise de

Modos de Falhas, Efeitos e Fatores Críticos (Failure Modes, Effects and Criticality Analysis - FMECA); Análise de Árvore de Falhas (software Fault Tree Analysis - SFTA); e Redes Petri Temporais.
O emprego da Técnica de Análise de Árvore de Falhas de software (software Fault Tree Analysis - SFTA) envolve a construção de um diagrama lógico, mostrando a provável seqüência de falhas que leva a um determinado Estado Inseguro [Ippolito e Wallace, 1995].
A Técnica de Análise de Modos de Falha, Efeitos e Fator Crítico (Failure Mode, Effects and Criticality Analysis - FMECA) envolve a utilização do raciocínio indutivo para determinar o efeito no Sistema da falha de um componente em particular, incluindo instruções de software [Ippolito e Wallace, 1995].
Pode também utilizar-se a técnica Análise de Rede Petri-Temporal [Leveson e Stolzy, 1987], à qual cria uma representação gráfica do Sistema, que pode ser inspecionada para a determinação de falhas relacionadas a tempo.

As atividades pertencentes à Análise de Insegurança de Sistema visam identificar Estados Inseguros relacionados à interação do computador com outros componentes do Sistema. Para tanto, propõe-se o desenvolvimento de uma SMFE, para integrar os resultados das técnicas FMECA e SFTA.
A utilização conjugada de SFTA e FMECA permite analisar as causas e os efeitos de cada falha do software, armazenando essas informações em uma tabela denominada Sumário de Modos de Falhas e Efeitos (SMFE) [Moura, 1996].
Finalmente, realiza-se a Análise de Insegurança na Operação e Suporte para identificar os Estados Inseguros, o uso e a manutenção do Sistema. Não propõe-se nenhuma técnica específica para esta Análise, podendo realizá-la a partir da investigação do SMFE.
Após a identificação dos Estados Inseguros, deve-se avaliá-los e classificá-los de acordo com o seu Fator Crítico. Para tanto, considera-se fatores como Severidade e Probabilidade de ocorrência do acidente associada a cada Estado Inseguro, como exposto no Item 4.
No final deste procedimento, deve-se avaliar os Estados Inseguros não evitados, controlados ou recuperados completamente, visando garantir que os mesmos não representem uma ameaça à Segurança de software.
A Figura 3, a seguir, mostra uma síntese da Sistemática para Análise de Segurança de software.

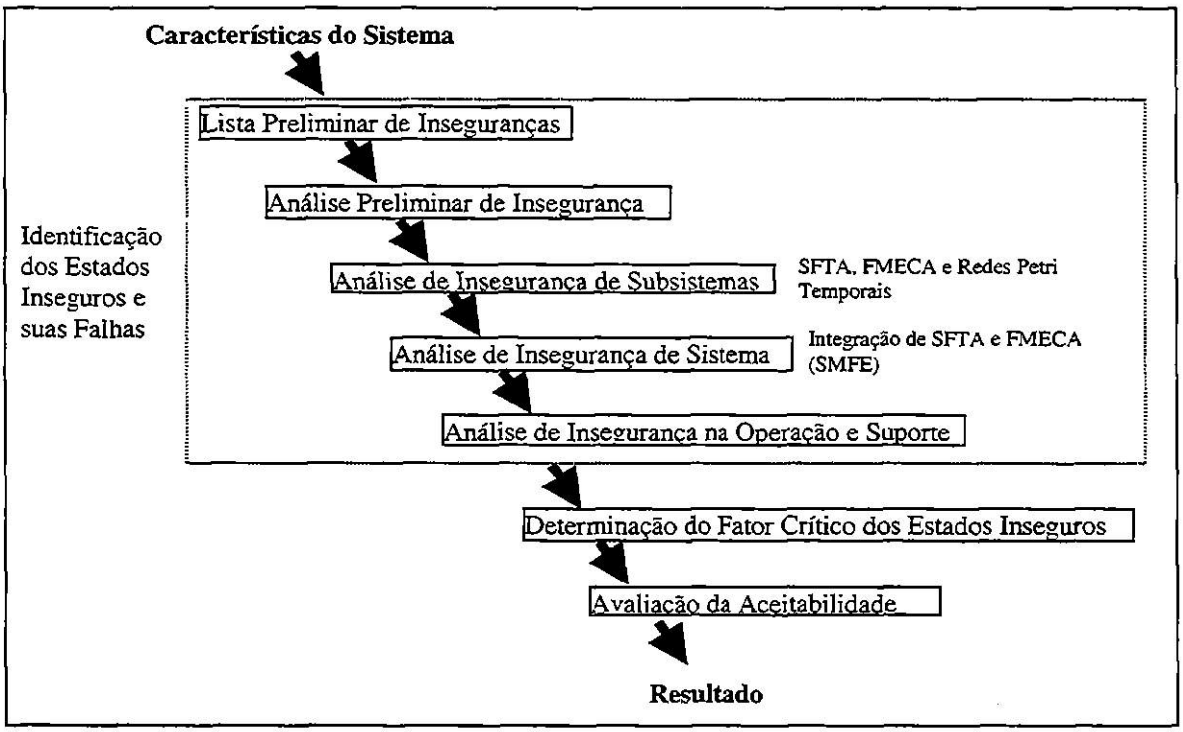


Figura 3: Sistemática para Análise de Segurança de Software.

A figura acima sumariza a Sistemática para Análise de Segurança de SCEA, apresentando toda seqüência de atividades e técnicas constituintes desta análise. Ressalta-se que a realização destas análises é iterativa, reavaliando seus resultados à medida que as informações sobre o sistema aumentam.

7. Conclusões e Recomendações

Apesar da crescente importância dos computadores na sociedade moderna e de sua integração aos Sistemas Aeroespaciais, ainda não se tinha notícia da existência de uma Sistemática eficaz para a Análise de softwares Críticos Embarcados Aeroespaciais, capaz de auxiliar na obtenção de níveis apropriados de Segurança de software. A falta de uma Sistemática deste tipo, vinha aumentando a complexidade, o custo e o tempo de desenvolvimento destes softwares.

Neste artigo, apresentou-se uma proposta de Sistemática para Análise de Segurança de software Crítico Embarcado Aeroespacial (SCEA). Esta nova Sistemática possui diferentes atividades para identificação de Estados Inseguros e das falhas que os originaram, determinação do Fator Crítico, e ainda, avaliação da aceitabilidade dos níveis de Segurança do Sistema.

Dessa forma busca-se auxiliar no incremento do nível de Segurança de software exigida para este tipo de aplicação , desde as primeiras fases de seu desenvolvimento.

A realização criteriosa deste procedimento de Análise de Segurança de software, pode reduzir o esforço de detecção e correção de defeitos, reduzindo também, o esforço para o desenvolvimento do SCEA.

Pode-se considerar o uso da técnica Análise de Rede Petri Temporal facultativo, uma vez que, a mesma só apresenta resultados significativos em softwares com restrições severas em relação ao tempo e a sincronização, e nem todos os SCEA possuem estas características.

Recomenda-se também, o aprimoramento das ferramentas CASE e das técnicas para Análise de Segurança de software, permitindo melhor observar a interação do software com os outros componentes do Sistema.

Deve-se também, desenvolver Sistemáticas como a deste artigo, que se apliquem ao Projeto e à Implementação do SCEA. Busca-se assim, abranger todo o ciclo de desenvolvimento do software.

A curto e médio prazos, pretende-se aplicar esta Sistemática para Análise de Segurança de SCEA em projetos do ITA (Instituto Tecnológico de Aeronáutica), visando fornecer meios que possibilitem o acréscimo nos níveis de Segurança no desenvolvimento de SCEA's nesta instituição.

Posteriormente, pode-se utilizar a Sistemática proposta neste trabalho em disciplinas relacionadas a Engenharia de software e Qualidade de software, visando qualificar um número maior de profissionais para o desenvolvimento de SCEA.

8 Referências Bibliográficas

FERNANDES A. A. *Gerência de software Através de Métricas,* Garantindo a qualidade do Projeto, processo e produto. Atlas Ed., São Paulo, 1995.

GHEZZI C., JAZAYERI M., MANDRIOLI D. *Fundamentals of software Engineering*. Prentice-Hall Inc., New Jersey , 1991.

IPPOLITO L. M. , WALLACE D. R. *A Study on Hazard Analysis in High Integrity software Standards and Guidelines*. National Institute of Standards and Technology. Disponível por meio da www no endereço http://hissa.ncsl.nist.gov/ HHRFdata/Artifacts/ITLdoc/5589/hazard.html, Janeiro de 1995.

LEVESON N. G. *software Safety in Embedded Computer Systems*. Comunications of the ACM, Fevereiro de 1991, p. 35 - 45.

LEVESON N. G. , HARVEY P. R. *Analyzing software Safety*. IEEE Transactions on software Engineering , Setembro de 1983, Vol. SE-9, n° 5.

LEVESON N. G., STOLZY J. L. *Safety Analysis Using Petri Nets*. IEEE Transactions on software Engineering , Março de 1987, Vol. SE-13, n° 3.

LOVISI FILHO E., CUNHA A. M. *Uma Abordagem para o Desenvolvimento de software Crítico Embarcado Aeroespacial com Garantia de Segurança*. S. José dos Campos, In: Anais do Simpósio sobre Segurança em Informática, 1999. p. 57-67.

MINISTRY OF DEFENSE. *Safety Management Requirements for Defense System*. London, 1997. (Defence Standard 00-56). Disponível por meio da www no endereço http://www-scm.tees.ac.uk/ hazop/html/56.htm.

MONTALK J. P. P. *Computer software in Civil Aircraft*. London: Butterworth-Heinemann, 1993, volume 17, n° 1. p. 17-23.

MOURA C. A. T. e SANTELLANO J. "*software Aeroespacial no Brasil:* Um pequeno Balanço e perspectivas para o Setor". A publicar, 1999.

MOURA C. A. T. , SANTELLANO J. , NETO A. A. *software e Segurança de Sistemas Aeroespaciais*. In: Anais do encontro de Iniciação Científica e Pós Graduação, Outubro de 1996, p. 166-170.

MOURA C. A. T. *Uma Estratégia de Análise de Segurança de software para aplicações Críticas*. S. José dos Campos: ITA, 1996. Tese de Mestrado.

PARNAS D. L. *Can software for the Strategic Defense ever be error free?*. Computer, Novembro de 1986, p. 61 - 67.

PARNAS D. L. *software Aspects of Strategic Defense Systems*. American Scientist, Setembro/Outubro de 1985, vol. 73, p. 432 - 440.

PÔRTO I. J., DE BORTOLI L. A. *Sistemas Tolerantes a Falhas*. Disponível por meio da www no endereço http://www.inf.ufgrs.br/ gpesquisa/tf/portugues/ensino/lisangela/segsoft. html, Julho de 1997.

PRESSMAN R. S. *Engenharia de software*. Quarta Edição, Makron Books, New Jersey, 1995.

SANTELLANO J. , MOURA C. A. T. , LIMA A. C. et al. *Estratégias de Segurança de software:* A Abordagem do Padrão MIL-STD-498. A publicar, 1998.