SISTEMAS CRÍTICOS, UM GUIA PARA ELICITAÇÃO DE REQUISITOS DE *SOFTWARE*

MACEDO, Paulo Cesar de
Faculdade Santa Lúcia
pcmacedo@me.com

CATINI, Rita de Cássia
Faculdade Santa Lúcia
ritacatini@gmail.com

CATINI NETO, Carlos
ETEC Pedro Ferreira Alves
catini.it@gmail.com

RESUMO

Os projetos de software considerados de segurança crítica requerem muita atenção das empresas desenvolvedoras. Existem inúmeras lacunas a serem pesquisadas com relação a metodologias a serem aplicados nesse tipo de sistema, além das diferenças na forma de conduzir projetos. Assim, a engenharia de software enfrenta os problemas de responsabilidades que sistemas desse tipo exigem. O objetivo deste artigo é apresentar um Guia denominado G2S para o acompanhamento de projetos de sistemas considerados críticos a segurança. Como método utilizamos uma pesquisa em empresas do segmento e como resultado um conjunto de boas práticas para evitar falhas em sistemas onde vidas ou grandes somas de dinheiro podem ser perdidas.

PALAVRAS-CHAVE: *sistemas críticos; guia; safety; metodologias.*

INTRODUÇÃO

Os sistemas críticos se fazem presentes em muitas coisas do nosso

Recebido em 11-04-2015 • Aceito em 25-04-2016

cotidiano, desde uma simples viagem de automóvel ou de trem, ou ainda uma consulta de rotina à um médico, você pode estar sendo submetido a um controle crítico. Sua vida pode estar em risco, seja por um diagnóstico interpretado por um sistema de maneira errônea, um acidente ocasionado por falta de informação dos sensores de um carro ou a falha na interação de um condutor de trem, por exemplo.

Os sistemas críticos devem ser minuciosamente analisados e projetados com qualidade, pois vidas humanas ou ambientes são expostas a ele. Uma falha nesses sistemas, por menor que seja, pode trazer problemas irreparáveis. Portanto, cabe aos engenheiros ou arquitetos de *software* desenvolver e utilizar mecanismos para garantir o seu funcionamento.

Muitas empresas adotam seu próprio método de análise, nos quais definem formas das mais variadas, fazendo de cada projeto um método específico, onde aquela realidade demanda uma forma diferente de trabalho.

Sommerville (2011), por razões que não são óbvias, afirma que algumas vezes os sistemas computacionais entram em colapso e falham em disponibilizar os serviços que foram pedidos.

O objetivo desse artigo é apontar o funcionamento de alguns métodos de análise a sistemas críticos e propor um guia denominado G2S , mostrando quais passos se fazem necessários para resolver problemas, bem como introduzir a noção de confiança e a importância das quatro dimensões que alicerçam os sistemas críticos: disponibilidade, confiabilidade segurança e proteção.

Na análise de sistemas críticos é preciso evitar enganos durante o desenvolvimento. Caso o sistema já esteja em uso é fundamental conseguir detectar e eliminar os possíveis erros antes que esses ocorram. Atualmente, no Brasil existem poucos estudos nesse segmento, que é embasado, quase que em sua totalidade, em artigos de outros países. Um dos motivos prováveis é o fato das empresas que exploram controladores, microprocessadores e as linguagens de programação, não serem nacionais.

Realizar um guia é um fato motivador para elaboração desse artigo e mostrar como algumas alternativas contribuem de uma forma ou de outra também se faz necessário.

Este artigo está dividido em seções que levarão o leitor a uma noção de como os sistemas críticos estão sendo utilizados hoje e aponta tendências para futuras pesquisas nesse segmento.

Nos primeiros capítulos, após uma introdução do tema, o artigo mostra

Recebido em 11-04-2015 • Aceito em 25-04-2016

um breve histórico de quando e como surgiu esse nível de preocupação, bem como alguns pontos de vista dos autores mais conceituados no assunto.

Posteriormente serão abordados os fundamentos e conceitos relacionados ao tema, abordagem técnica e citações seguido do guia (G2S) proposto, contendo detalhes de sua utilização. Finalmente, uma conclusão do trabalho será descrita com destaque às contribuições que os estudos apontaram e serão propostas análises e perspectivas para trabalhos futuros.

2. RISCOS E SISTEMAS DE SEGURANÇA CRÍTICA

A norma AS/NZS 4360 (2004) é destinada à inclusão e à importância da Gestão de Riscos na filosofia, nas práticas e nos processos de negócio de uma organização. Ainda que os conceitos de risco sejam repetidas vezes apontados em termos de perigo ou impacto negativo, a norma assiste os riscos como exibição às consequências da incerteza ou como possível desvio do que foi planejado ou do que é esperado.

Segundo Freitas e Gomez (1997), em nosso dia a dia, e cada vez mais, nos defrontamos com notícias referentes aos riscos que determinadas tecnologias, na forma de produtos ou processos industriais, podem causar às pessoas e aos lugares que vivem.

Algumas áreas apresentam os sistemas críticos como principal fator de preocupação. Seguem algumas delas:

a) Aeronáutica, de acordo com DIVOP(2003)

As atividades de busca e prevenção de acidentes aeronáuticos, no Brasil, ocorrem desde a década de 20. Com o aparecimento da aviação militar, tanto na Marinha quanto no Exército, as investigações dos acidentes ou incidentes aeronáuticos procuravam sempre a apuração de culpa, por meio de inquérito.

Depois de criar o Ministério da Aeronáutica, na década de 40, as averiguações e inquéritos de acidentes foram integradas sob os cuidados Inspetoria Geral da Aeronáutica, e passaram a sofrer um processo de constante evolução.

Segundo Gideon (2004), os procedimentos de segurança (*safety*) das empresas, sejam elas de aviação ou não, existem por várias razões. Alguns deles são requisitos exigidos por lei e determinam a forma como o avião ou outros veículos de transporte, que envolvam vidas, serão desenvolvidos, nas suas diversas fases (projeto, construção, manutenção e operação).

Recebido em 11-04-2015 • Aceito em 25-04-2016

b) Férrea, de acordo com Cauduro (2002)

Já nas linhas férreas como as do Metrô, Fundação da Companhia do Metropolitano de São Paulo foi fundado na década de 60, os primeiros controles de trafego eram manuais e se fazia necessária a intervenção humana na maioria dos controles elétricos. Acidentes eram normalmente relatados como culpa de fatores operacionais, onde falhas humanas eram comuns.

c) Automóveis, segundo Filgueiras (2006)

Ao longo do tempo os sistemas foram evoluindo de forma que chegaram até aos veículos, e a partir do sistema de frenagem, combustível e motor, ganharam dimensões importantes, fazendo com que o uso de dispositivos computáveis fosse um diferencial entre o mercado dos mesmos.

Um carro equipado com um computador de bordo oferece ao proprietário uma interface de informações do estado do veículo, importante principalmente para motoristas que não entendem dos princípios da mecânica. Uma informação de falta de óleo, ou superaquecimento, ou de um simples vazamento de ar dos pneus pode salvar uma vida, evitando um possível acidente.

d) Medicina, de acordo com Filgueiras (2006)

Na área médica, equipamentos inteligentes podem colaborar com diagnósticos e tratamentos, onde sensores miniaturizados permitem desenvolver sistemas automatizados para administração de remédios e controles de temperatura, batimentos cardíacos e transfusão sanguínea. Inicialmente, todo processo de tratamento de doenças era de total responsabilidade dos médicos e, muitas vezes, por falta de informações acabavam por não saber o histórico do paciente e aplicavam a medicina de modo falho.

Com o uso de computadores, sejam nos processos ou no simples arquivamento das informações, a área médica, sem dúvidas, é uma das afetadas por problemas de sistemas críticos. Segundo Leveson, Mats e Jon (1999), os defeitos de *software* causam falhas de *software* quando o código com defeito é executado com um conjunto de entradas que o expõe ao defeito. Muitos acidentes aconteceram envolvendo erros na área médica, porem mostrá-los não é o foco desse trabalho.

Os autores ainda afirmam que a especificação de sistemas críticos originou-se a partir do momento em que a confiabilidade nos sistemas foi

Recebido em 11-04-2015 • Aceito em 25-04-2016

colocada em cheque. A união entre sistemas elétricos, mecânicos e informáticos envolvendo vidas humanas veio à tona, já que acidentes não poderiam ter como culpados e responsabilizados apenas operadores. Quando sistemas foram considerados culpados e seus realizadores julgados, todos os sistemas passaram a serem vistos com outros olhos e preocupações.

Atualmente, pessoas de diversos países estão envolvidas no meio acadêmico em buscar soluções práticas para colaborar com essa análise. Existem várias técnicas que foram propostas como abordagem para esse estudo, elas incluem revisões e *checklists*, além de técnicas mais formais, como análise de rede de Petri, a lógica formal de Jahanian e Mok (1986) e a análise de árvore de defeitos de Leveson e Harvey (1983), porém consideradas ultrapassadas haja vista o cenário tecnológico atual.

Um sistema considerado crítico é um sistema em que as falhas podem resultar em perdas econômicas significativas ou ameaças à vida humana. (SOMMERVILLE, 2011)

As palavras em inglês *Safety* e *Security* são, em geral, traduzidas ambas para segurança, mas em sistemas críticos, elas significam coisas diferentes.

Neste artigo vamos usar o termo Safety que segundo Leveson, Mats e Jon (1999), pode ser definida como a ausência de acidentes. Acidentes, por sua vez, são eventos indesejados que podem envolver a perda de vida humana, danos ao meio ambiente ou danos materiais expressivos. Riscos de acidentes ou incidentes devem acontecer mesmo nas melhores intenções mesmo quando o sistema está trabalhando corretamente, segundo o esperado. Além disso, um *software* pode falhar, ou interromper seu funcionamento em um estado considerado seguro, de forma que a segurança não significa garantia de confiabilidade.

Sistema seguro é aquele que está livre de acidentes ou perdas e é um atributo que reflete a capacidade do sistema de operar, normal e anormalmente, sem ameaçar as pessoas ou o ambiente (SOMMERVILLE,2011).

É praticamente impossível atingir total e completa segurança, mas procura-se chegar o mais próximo possível do ideal. Sommerville (2011) cita ainda a confiança como fator primordial e apresenta quatro dimensões a serem compreendidas:

- Disponibilidade – é a probabilidade de que o sistema estará funcionando e em condições de disponibilizar serviços úteis a qualquer momento;
- Confiabilidade – é a probabilidade, por determinado período de tempo, de que o sistema disponibilizará serviços de maneira correta, conforme o usuário espera;

Recebido em 11-04-2015 • Aceito em 25-04-2016

- Segurança – é um julgamento da probabilidade de o sistema causar danos às pessoas ou a seu ambiente;
- Proteção – é um julgamento da probabilidade de um sistema poder resistir à invasão acidental ou deliberada.

Sendo assim, a confiança é fundamental para evitar que usuários se recusem a usar os sistemas ou usem de maneira errônea, causando perda de informações e retrabalho. Tentar recuperar a confiança é muito difícil, sem falar no fator custo, que pode ser elevado no reparo de sistemas.

Para Lemos e Saeed (2006), quando se considera a confiança em sistemas críticos, podem ser apontados três tipos de componentes de sistemas sujeitos a falhas: *Hardware*, *software* e operadores. Ainda segundo os autores Lemos e Saeed (2006), nota-se que a base essencial para o desenvolvimento de *software* de segurança crítica é estabelecer uma alta qualidade na especificação dos requisitos e indica que temos que modelar, analisar e documentar todos os passos de desenvolvimento bem como seus *links* (correlação) .

A análise de segurança do processo determina se o risco associado com a especificação é aceitável ou não.

Os requisitos de confiabilidade devem ser definidos quantitativamente na especificação dos requisitos do sistema. Leveson, Mats e Jon (1999), indicam que existem várias métricas diferentes de confiabilidade e as especificações não funcionais da mesma podem levar a criação de novos requisitos funcionais no sistema, a fim de reduzir o número de falhas e, portanto, aumentar sua confiabilidade.

A análise de perigos consiste, segundo Pressman (2011), no processo de especificação de segurança, e envolve identificar condições perigosas que possam comprometer a segurança do sistema e a análise de risco que, segundo o autor, é o processo de avaliação da probabilidade de que um perigo possa resultar em um acidente.

O fator erro, segundo Filgueiras (2006), é a diferença entre o comportamento desejado, esperado e especificado do mundo real. O erro humano é o resultado natural do processo para solução de problemas e como o ser humano é parte de um sistema, que na computação é composto por *hardware*, *software* e processos. Esse erro pode ocasionar a disfunção ou falhas (*failure*) do sistema, ou seja, não cumpre seus requisitos.

Após o sistema pronto é comum as empresas utilizarem alguns métodos e ferramentas para verificar o quão seguro ele está, baseando-se em suas falhas. Para Ericson (2005) e Yang (2007) podemos considerar como principais métodos:

Recebido em 11-04-2015 • Aceito em 25-04-2016

- *Fault Tree Analysis (FTA)* - Trata-se de uma representação gráfica de relações lógicas entre eventos de falhas (uma árvore de falha), onde o evento topo é ramificado em eventos contribuintes por meio da análise de causa-efeito;
- *Event Tree Analysis (ETA)* - Análise de árvore de eventos é uma técnica de análise para identificar e avaliar a sequência de eventos em um cenário de potencial acidente após a ocorrência de um evento inicial;
- *HAZard and OPerability Studies* (*HAZOP)* - Estudo de Perigos e Operabilidade é uma técnica para identificar e analisar perigos e preocupações operacionais de um sistema. A Análise HAZOP procura por perigos resultantes do potenciais desvios na objetivo do projeto operacional;
- *Failure Modes and Effects Analysis* (*FMEA)* - A Análise do Modo e Efeitos de Falha é uma técnica usada para definir, identificar, e eliminar falhas conhecidas ou em potencial antes que alcancem o cliente;
- *Preliminary Hazard Analysis (PHA)* - Análise de risco preliminar, Geralmente a primeira análise rigorosa que é realizada para identificar os riscos, fatores de perigo, percalços e risco ao sistema.

Além dos métodos listados, ainda podemos destacar outros de menor expressão, mas que ainda merecem ser citados, segundo Stamatis (2003) como: *What-if analysis*, Incidentes Críticos (TIC) e *Reliability Block Diagrams* (RBD).

Todos os métodos citados contribuíram de alguma forma para realização do guia proposto e serviram de base para indicação dos testes obrigatórios desde o início do projeto, além de contribuir metodologicamente como podemos observar no capítulo a seguir.

3. METODOLOGIA

Para realização desse artigo além da pesquisa bibliográfica foram utilizadas informações provenientes de 32 empresas desenvolvedoras que enfrentam constantemente problemas de responsabilidade e contrato. Essas empresas foram escolhidas por terem características em comum, por exemplo: Todas trabalham no desenvolvimento de *software* para outras empresas, usam algum tipo de metodologia em seus projetos e conduzem projetos com equipes com mais de 10 pessoas envolvidas, de forma direta ou indireta.

Não obtivemos autorizações para divulgação dos nomes dessas empresas devido possíveis críticas e geração de desconfianças em seus mercados frente aos competidores, além de algumas que não quiseram responder a

Recebido em 11-04-2015 • Aceito em 25-04-2016

pesquisa. Sendo assim, foi utilizado um questionário de perguntas abertas como instrumento para captar as informações relativas a esses problemas. Os dados foram coletados de 26 de janeiro à 24 de março de 2015 por meio da Internet, em um formulário disponível em: https://docs.google.com/forms/d/12wVzUb2iL6NTaDo0_d6wl4ZlhfQyFIHbqEFBb8P4IYg/viewform.

Responderam ao formulário 32 (trinta e dois) profissionais que atuam no desenvolvimento de *software*, e que em algum momento atuaram em projetos de sistemas considerados críticos no Brasil. Outras respostas serão tabuladas para outros artigos nesse mesmo segmento, ou seja, as demais respostas às perguntas somarão em trabalhos futuros

O questionário aplicado possui 5 (cinco) perguntas sobre o negócio e 10 (dez) perguntas objetivas sobre como controlam seus processos durante o desenvolvimento de um projeto de *software*, a saber:

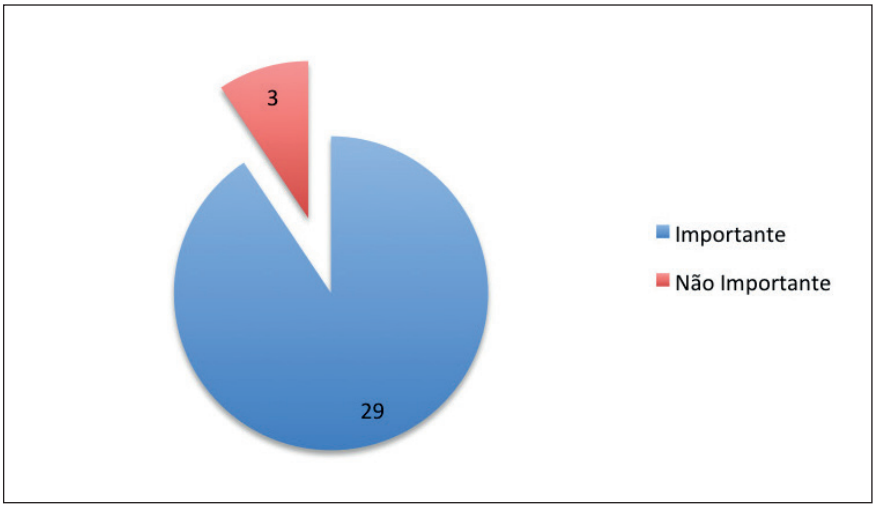
1. Quando estabelecem a primeira entrevista com o cliente (*Stakeholder*), o analista relaciona a criticidade do projeto?
2. É feito uma análise dos riscos iniciais do projeto? Se sim, de que forma?
3. A empresa desenvolvedora aplica testes desde o início, na elicitação dos requisitos? Se sim, quais testes?
4. Existe uma equipe específica para testar o sistema diferente da que desenvolveu?
5. Utiliza a UML para estabelecer a modelagem dos sistemas propostos? Se sim, quais diagramas?
6. Realiza o procedimento de "Impacto" nos "Builds" durante a implantação? Se sim, com qual frequência?
7. Já enfrentou problemas jurídicos devido a falta de cuidados com sistemas críticos?
8. Exige a presença de pelo menos um representante da empresa (cliente) durante o desenvolvimento do sistema?
9. Documenta e atualiza todas as modificações ocorridas nos sistemas através da abertura de chamados em suporte?
10. Considera importante a criação de um guia para apoiar os gerentes e analista durante o desenvolvimento de sistemas críticos?

Após o recebimento das respostas pôde-se tabular os resultados e elencar pontos importantes que motivaram a criação do guia G2S. A seguir, alguns desses pontos serão apresentados de forma a justificá-los quantitativamente.

Como relação à questão 10, ficou evidenciada a necessidade de um guia, como podemos observar na **Figura 1:**

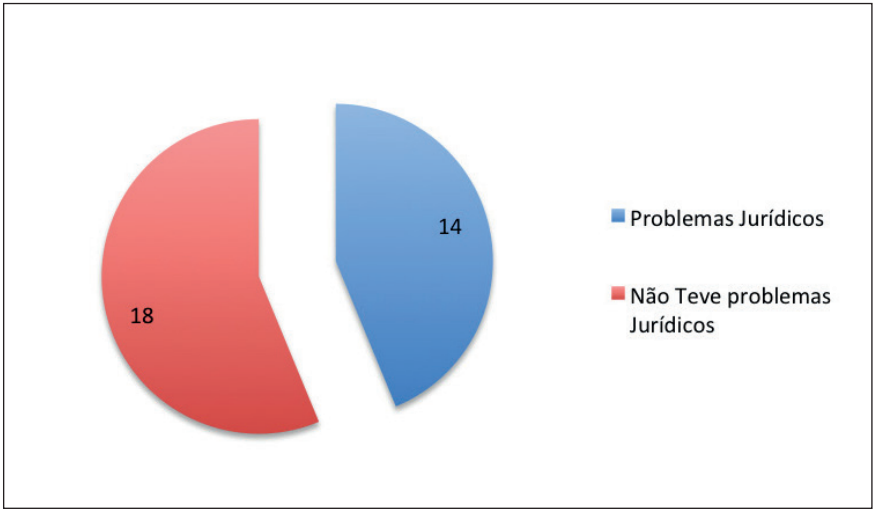
Recebido em 11-04-2015 • Aceito em 25-04-2016

Figura 1 – Importância do guia



Com relação à problemas jurídicos ou perda de contrato devido a responsabilidades (questão 7), foram detectados um número considerável (**Figura 2)** como pode-se observar 14(quatorze) empresas já enfrentaram problemas jurídicos:

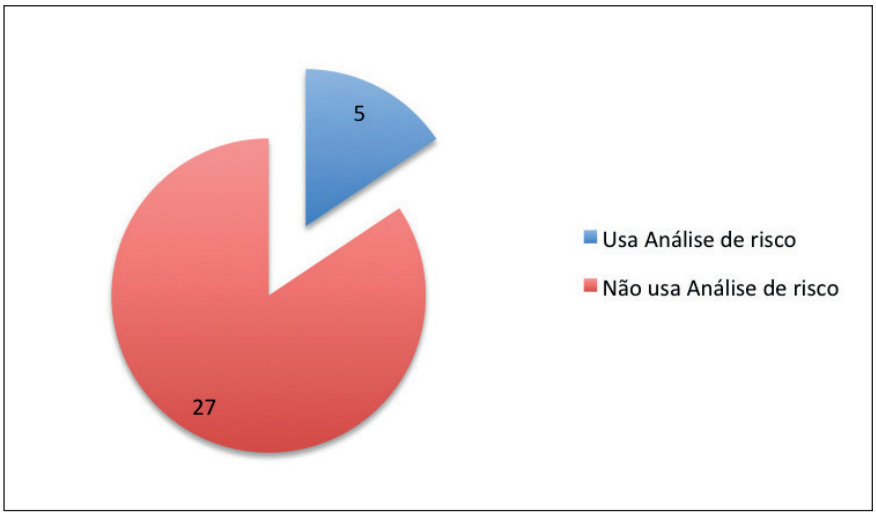
Figura 2 – Problemas jurídicos ou de responsabilidade



Recebido em 11-04-2015 • Aceito em 25-04-2016

Com relação à análise de Risco (questão 2) das empresas desenvolvedoras analisadas, a diferença foi significativa no sentido de não utilizar nenhuma técnica para analisar os riscos eminentes que podem acontecer independente de projeto, o (**Figura 3)** ilustra o resultado obtido onde 27 empresas relataram não utilizar:

Figura 3 – Análise de Risco



Nem todas as respostas às perguntas foram tabuladas, portanto não possuem gráficos comparativos, o motivo é que as resposta serão utilizadas para a evolução do guia em trabalhos futuros.

3.1 O guia G2S

A seguir o guia proposto contendo os elementos fundamentais e suas respectivas justificativas de uso.

Recebido em 11-04-2015 • Aceito em 25-04-2016

GUIA PARA ELICITAÇÃO DOS REQUISITOS DE SOFTWARE EM SISTEMAS DE SEGURANÇA CRÍTICA – G2S	
I. Captação inicial dos requisitos O analista ao visitar a empresa ou decidir por fazer um sistema deve lançar mão dos instrumentos de coleta de dados convencionais tais como: Entrevista, questionários e formulários; Em qualquer um dos casos os mesmos devem ser descritos em forma de documento em formulário próprio, timbrado e devidamente assinados pelas partes envolvidas (Analista de Sistemas e *Stakeholders*); Várias visitas podem ser feitas e o documento atualizado até uma versão final. Estes dados são transportados para fichas de requisitos divididas em funcionais e não funcionais, em alguns métodos separados em estórias descrevendo o requisito, e de qualquer forma devem constar os dados dos solicitantes, bem como datas e assinaturas. As fichas ou estórias devem ser enumeradas afim de proporcionar a rastreabilidade dos requisitos. A partir dessas fichas o analista ou gerente de projeto pode criar a lista de riscos de projeto coletando alguns riscos específicos como: a) Riscos de Relacionamento; b) Riscos Organizacionais; c) Riscos Gerenciais; d) Riscos Financeiros; e) Riscos Técnicos ; f) Riscos Legais; e, g) Riscos de Planejamento.	**Justificativa:** Em sistemas de segurança crítica aquilo que foi desenvolvido deve ser compatível e representar o que foi solicitado, ficando a responsabilidade a cargo do solicitante (*Stakeholder*), desta forma, as fichas devem ser redigidas em forma de contrato, a fim de evitar problemas de ordem judicial futuros. **Artefatos:** Entrevista, Questionários, Formulários atas, Fichas de Requisitos/Estórias e Planilhas/ Lista de riscos de projeto.
II. Modelagem Independente da metodologia escolhida (ágil ou tradicional) e considerando que todas exigem que as informações sejam modeladas afim de facilitar o entendimento das equipes de desenvolvedores, nesta fase são feitos as representações gráficas, os diagramas que ilustram o que foi solicitado. Esses diagramas devem ser enumerados e referenciados nas fichas de requisitos, ou seja, deve constar nas fichas ou em uma lista paralela os requisitos e seus diagramas correspondentes. Algumas metodologias exigem alguns diagramas, mas se tratando de sistemas de segurança crítica, todos os diagramas podem ser necessários. Basicamente os diagramas da UML(linguagem comumente usada para modelagem) listados a seguir são imprescindíveis: a) Diagrama de Classes; b) Diagrama de Casos de Usos; c) Diagrama de Sequência; d) Diagrama de Atividades; e) Diagrama de Estados; f) Diagrama de Componentes; e, g) Diagrama de Implantação. Inspeções de conformidade entre diagramas e requisitos são necessários nessa fase.	**Justificativa:** Em sistemas onde vidas ou finanças estão comprometidas o fator "rastreabilidade" pode ser fundamental para localizar a origem da solicitação. Usar os números das fichas e dos diagramas podem facilitar o rastreamento principalmente quando o sistema já foi implementado. **Artefatos:** Diagramas da UML e Relatório de inspeções.

Recebido em 11-04-2015 • Aceito em 25-04-2016

GUIA PARA ELICITAÇÃO DOS REQUISITOS DE SOFTWARE EM SISTEMAS DE SEGURANÇA CRÍTICA – G2S	
III. Protótipo de telas – Design de interfaces Neste momento deve-se desenhar as telas do sistema e imprimi-las em forma de documento para que o *Stakeholder* (solicitante) faça a validação das mesmas, ou seja, o analista novamente vai conferir junto ao solicitante se o que ele pediu será devidamente implementado. Pode ocorrer nesta fase de nem todas as telas estarem desenhadas e de algumas darem apenas uma ideia ao solicitante de como será feito. Isso implica que trata-se apenas de um protótipo e nada definitivo. Mais uma vez utilizaremos a numeração para identificar cada tela e atualizar a lista ou as fichas com esse número.	**Justificativa:** Nesse tipo de sistema não podemos desenvolver protótipos do mesmo para testes funcionais, apenas modelos de telas. Desta forma, não colocaremos o sistema inacabado e com possíveis erros para operar com informações reais, pois uma falha acarreta em perdas significantes. **Artefatos:** Telas/Interfaces do sistema.
IV. Implementação O sistema pode ser desenvolvido de forma iterativa (em partes) e incremental, mas suas entregas (*release*) devem ser completas e testadas. Os testes podem ser automatizados afim de agilizar o desenvolvimento. Uma equipe diferente daquela que codificou o sistema será imprescindível, evitando assim que os testes sejam tendenciosos. Os seguintes testes funcionais (caixa preta) são obrigatórios para sistemas desse tipo: a) Teste unitário (para cada funcionalidade); b) Teste tratamento de erros; c) Teste de controle; d) Teste de paralelo; e, e) Teste de suporte manual (sensível ao contexto). E os testes estruturais (caixa branca) a seguir também devem ser observados: a) Teste de estresse (capacidade do sistema); b) Teste de execução (comportamento do sistema); c) Teste de recuperação (contingência); d) Teste de operação; e) Teste de conformidade; e, f) Teste de segurança.	**Justificativa:** Um sistema testado desde o início evita o retrabalho e diminui os custos com manutenção. Os testes podem ser realizados desde o início, mesmo durante a fase de captação dos requisitos, conforme citado anteriormente nesse guia. **Artefatos:** Resultados dos testes em listagens, Scripts de testes.
V. Implantação Para algumas metodologias que fazem entregas constantes de versões do *software* para o cliente, é importante, no caso dos sistemas críticos, que seja observada e monitorada a utilização inicial, ou seja, no momento em que a nova versão for "impactada" (nome popular da nova instalação) o analista tem que estar preparado para retornar	**Justificativa:** Nenhum software está livre de defeitos, pode ocorrer algumas falhas pontuais, mas na medida do possível deve ser possível

Recebido em 11-04-2015 • Aceito em 25-04-2016

GUIA PARA ELICITAÇÃO DOS REQUISITOS DE SOFTWARE EM SISTEMAS DE SEGURANÇA CRÍTICA – G2S	
a versão anterior não permitindo que o problema encontrado traga grandes danos. Pode ser necessário fazer testes em paralelo onde o software é submetido por várias vezes a situação de uso e seus resultados são observados. Caso haja algum erro ou problema, o mesmo não deve ser instalado antes da correção.	reverter as situações logo no primeiro momento de uso. **Artefatos:** Controle de versões e Resultados dos testes realizados.
VI. Manutenção Os problemas encontrados pelos usuários devem ser documentados e solucionados o mais breve possível. Neste documento além do problema descrito na abertura do chamado deve-se constar também a solução adotada, bem como as assinaturas dos responsáveis pelo chamado. Alguns itens podem ajudar a evitar problemas em sistemas futuros.	**Justificativa:** Uma alteração no código pode gerar problemas e conflitos com o que foi solicitado e o sistema não conferir com o que foi inicialmente solicitado gerando uma divergência. **Artefatos:** Documento de chamados a suporte.
VII. Encerramento do projeto Após o sistema estar concluído e não haja mais alterações a serem feitas, um documento deve ser redigido constando todas as características do *software*, Atas de reunião, descrição sucinta das funcionalidades bem como um parágrafo isentando a responsabilidade da empresa quanto ao mal uso do sistema criado. Desta forma, assim que finalizado, a empresa estará livre para novos desafios e não obrigada a prestar contas sobre o sistema já desenvolvido. Pode ser necessário a empresa mantem um departamento jurídico para resolver essas questões.	**Justificativa:** A empresa posteriormente pode usar mecanismos de má fé para sonegar impostos, venda de dados confidenciais, crimes cibernéticos, entre outros. Cabe a empresa desenvolvedora se precaver evitando indiciamentos. **Artefatos:** Termo de encerramento, Versão final do contrato de aquisição de software.

CONSIDERAÇÕES FINAIS

Após a pesquisa podemos observar a grande necessidade das empresas em ter um modelo de como lidar com sistemas críticos. Ficou evidenciado, por meio das respostas do questionário, que as mesmas já tiveram problemas, inclusive jurídicos, por não submeterem seus sistemas

Recebido em 11-04-2015 • Aceito em 25-04-2016

à uma análise crítica . Auxiliar as empresas nesse sentido comprova e justifica a necessidade deste trabalho, fazendo com que as mesmas possam se organizar adaptando-se a essa realidade. Podemos concluir que os sistemas críticos precisam ser construídos com total segurança, com muita atenção e que os testes são fundamentais. Com as respostas, ficou clara a necessidade do G2S, e isso motivou ainda a realizar pesquisas futuras sobre o assunto, que provavelmente nos darão ideias de como complementar novas versões do mesmo. Os resultados da pesquisa, bem como o referido guia, serão encaminhados às empresas que colaboraram, e desta forma podemos ainda captar como o guia se saiu em cada uma delas.

REFERÊNCIAS

AS/NZS 4360. Australian and New Zealand Standard, **Risk Management, Standards** . Australia/Standards New Zealand, 2004.

CAUDURO, J.C.. **Planejamento visual urbano: o sistema do metrô de Sâo Paulo**. Vol.1,2 e 3. 2002. Tese (doutorado) – Universidade de São Paulo, Faculdade de Arquitetura e Urbanismo.

DIVOP. **Divulgação Operacional**. Expediente utilizado para a divulgação de assunto de interesse da prevenção de Acidente Aeronáutico. República Federativa do Brasil - Comando da Aeronáutica - Departamento De Aviação Civil - Subdepartamento de Infraestrutura, 2003.

ERICSON, C. A.. **Hazard Analysis Techniques for System Safety**, Hoboken, New Jersey: John Wiley & Sons, 2005.

FILGUEIRAS, L. V. L.. **Human-Centred Design: using ISO as a reference (accepted).** In: 2006 ATRS Conference, 2006, Nagoya, 2006.

FREITAS C.M. de, GOMEZ C.M.. Tecnological risk from the perspective of the social sciences. História, Ciências e Saúde – Manguinhos, 1997.

GIDEON, Francis C.. **The Standard Handbook for Aeronautical and Astronautical Engineers** - Editora: McGraw-Hill, 2004

JAHANIAN ,F. ; MOK , A. K.. Safety analysis of timing properties in real-time systems - **Source IEEE Transactions on Software Engineering archive** - Volume 12 , Issue 9 (September 1986) table of contents - Special issue on reliability and safety in real-time process control Publisher IEEE Press Piscataway, NJ, USA , Year of Publication: 1986.

LEMOS, R.; SAEED , A.. **Safety Analysis Techniques for Validating Formal Models During Verification**, 2006.

LEVESON, Nancy G..HARVEY ,Peter R.. **Analyzing Software Safety**. IEEE Trans. Software Eng. 9(5): 569-579, 1983.

Recebido em 11-04-2015 • Aceito em 25-04-2016

LEVESON, Nancy G., MATS Per Erik H., JON Damon R.. **Designing Specification Languages for Process Control Systems**: Lessons Learned and Steps to the Future. ESEC / SIGSOFT FSE 99: 127-145, 1999.

PRESSMAN, R. S.. **Engenharia de Software**, McGraw-Hill, 7ª ed., 2011.

SOMMERVILLE, I.. **Engenharia de Software**. 9ª Ed; Pearson Addison. Wesley. São Paulo, 2011.

STAMATIS, D.H..**Failure mode and effect analysis: FMEA from theory to execution**, ASQC Quality Press. 2003.

YANG, Guangbin. **Life Cycle Reliability Engineering. Hoboken**, New Jersey: John Wiley & Sons, 2007.

Recebido em 11-04-2015 • Aceito em 25-04-2016