Assistente para Desenvolvimento de Software Crítico segundo a IEC 61508

Diego Bandeira, Taisy S. Weber, Sergio L. Cechin, Rodrigo Dobler, João C. Netto

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brazil
(taisy,dcbandeira,cechin,rjdobler,netto)@inf.ufrgs.br

Abstract. *An assistance tool can ease the development of critical software when it must follow a given safety standard as the IEC 61508. The standards are extensive and detailed making it a hard reading for software developers unfamiliar with the area of functional safety and fault tolerance techniques. We implemented a tool to help developers and testers to understand and apply each safety requirement of the IEC 61508 concerning the software life cycle. The tool verifies if the developers completed all the activities of a given phase of the life cycle and helps to keep the documentation needed for the certification process.*

Resumo. *Uma ferramenta de apoio ao projeto de software facilita o desenvolvimento de software seguro principalmente quando este deve seguir normas rigorosas de segurança crítica como a EC 61508. A norma é extensa e detalhada tornando penosa sua aplicação por desenvolvedores não familiarizados com as áreas de tolerância a falhas e segurança funcional. Apresentamos uma ferramenta de apoio para facilitar a aplicação de cada um dos requisitos de segurança da IEC 61508 levando em consideração o ciclo de vida de software crítico. A ferramenta verifica se todas as atividades foram completadas e mantém a documentação necessária ao processo de certificação de segurança.*

1 Introdução

Na aplicação de recursos computacionais para controle, automação e monitoramento na indústria nuclear e química, na exploração de petróleo e gás, na instrumentação médica e nos transportes, a necessidade de que os equipamentos programáveis executem corretamente as funções de segurança é evidente. Falhas no software desses equipamentos podem trazer prejuízos irremediáveis para a qualidade de vida ou danos irrecuperáveis ao meio ambiente. O termo segurança (*safety*) aplicado no contexto de prevenção de acidentes não deve ser confundido com o seu outro significado (*security*), que envolve proteção contra intrusos maliciosos, confidencialidade e integridade [1].

Exemplos de funções de segurança são: sinalização em ferrovias; controle de parada emergencial em indústrias químicas; bloqueio preventivo em maquinários pesados; sinais luminosos de alerta; sistemas de antitravamento de freios em automóveis; alarmes de incêndio e detecção de gases tóxicos. Funções de segurança costumavam ser implementadas com componentes discretos, não programáveis. Entretanto, é grande a demanda pelo emprego de componentes programáveis, tais como controladores lógicos e microcontroladores, com a função de segurança sendo executada por

Projeto parcialmente financiado pela Rede E3 – RIO-SIL/FINEP 01.11.0154.00

software [2]. As vantagens são inúmeras: reuso, maior flexibilidade, adaptabilidade e desempenho, além de um menor custo de desenvolvimento e manutenção.

Há muitas diferenças entre o processo de desenvolvimento de software em projetos tradicionais e em projetos de sistemas seguros. Nesses últimos, a criatividade do desenvolvedor é condicionada à aplicação de técnicas que já se mostraram adequadas na prática. Inovações tecnológicas não são bem acolhidas pelas normas de segurança, como a IEC 61508 [3], devido à falta de evidências sobre sua efetividade em situações reais de perigo. O ambiente de operação de sistemas relacionados à segurança é geralmente hostil. Fatores como temperatura, pressão, umidade, salinidade, gazes corrosivos, trepidação e impacto impõem um maior desgaste aos equipamentos, reduzindo sua vida útil e aumentando a probabilidade de apresentarem defeitos.

O maior desafio para um desenvolvedor de software seguro é seguir criteriosamente a regulamentação aplicável, escolhendo em cada fase do desenvolvimento as técnicas mais apropriadas para o domínio de aplicação. As normas são extensas e detalhadas e sua leitura e interpretação são difíceis. Para agilizar a sua aplicação, um software assistente passa a ser uma ferramenta essencial ao desenvolvedor [4]. A ferramenta direciona o desenvolvimento de maneira que nenhuma atividade ou documento importante seja ignorado. Ela acompanha passo a passo as tarefas do desenvolvedor e indica todas as atividades necessárias em cada fase do ciclo de desenvolvimento. Uma boa ferramenta mantém o histórico do projeto e facilita o rastreamento de alterações realizadas durante o período de desenvolvimento.

O objetivo do trabalho é implementar uma ferramenta de apoio para facilitar a aplicação da norma internacional IEC 61508 no desenvolvimento de software para a área de segurança funcional crítica. A ferramenta será aplicada no projeto RIO-SIL, que visa à produção de módulos com portas de entrada e saída digitais para sistemas instrumentados de segurança. Esses módulos são equipamentos completos de hardware e software conectados, por um lado, a barramentos de campo para comunicação com um ou mais controladores centrais e, do outro lado, conectados ao processo sendo instrumentado em tempo real. A ferramenta servirá não apenas aos propósitos do projeto RIO-SIL, mas terá um papel fundamental no treinamento da equipe de desenvolvimento. Será usada também como recurso de aprendizagem nas áreas de confiabilidade e tolerância a falhas para sistemas críticos.

A ferramenta não é um sistema crítico, nem executa funções de segurança. Por esse motivo seu desenvolvimento não está condicionado às restrições da norma IEC 61508. A ferramenta também não gera código seguro, nem executa os testes recomendados pela norma. A ferramenta apenas assiste na aplicação da norma, indicando ao desenvolvedor, a cada passo, as técnicas e métodos recomendados e mantendo em uma base de dados todos os documentos gerados necessários para a avaliação de um dado projeto.

O artigo apresenta o protótipo da ferramenta, focando nas fases relacionadas ao ciclo de vida de projeto de software. Inicialmente apresenta-se um resumo da norma IEC 61508 e a metodologia sugerida pela norma para o desenvolvimento de equipamentos seguros. Segue então um breve resumo de algumas ferramentas comerciais mais citadas e a apresentação ferramenta assistente de segurança, SVA, proposta. O artigo conclui com alguns comentários sobre a experiência na aplicação da norma e da ferramenta.

2 Padrão Internacional para Integridade de Segurança

Equipamentos relacionados à segurança atuam em ambientes onde acidentes podem provocar danos a pessoas ou ao meio ambiente. Tais equipamentos devem garantir confiabilidade, disponibilidade e integridade de segurança. Devem operar corretamente mesmo na ocorrência de falhas de hardware e software e de interferência externa. Caso não seja possível em alguma situação garantir a operação correta, o equipamento deve descontinuar sua operação alcançando um estado seguro sem provocar acidentes.

Normas, como a IEC 61508 [5], preconizam o uso de técnicas de prevenção e de tolerância a falhas para a redução de risco, além do projeto criterioso, documentado e auditável, tanto do hardware como de software, desde as primeiras fases do ciclo de desenvolvimento. O projeto de equipamentos relacionados à segurança exige sofisticado grau de elaboração e inúmeros cuidados técnicos. Além dos aspectos de engenharia de hardware e software, é preciso controlar também os fatores operacionais, os fatores gerenciais, o ambiente de operação e a manutenção. A norma IEC 61508 foi publicada originalmente em 2000. Em 2002 iniciou o processo de revisão [3]. Em abril de 2010 foi lançada a sua segunda edição, com base na qual este trabalho foi desenvolvido.

2.1 IEC 61508

A IEC 61508 [5] é um padrão para equipamentos elétricos, eletrônicos e eletrônicos programáveis relacionados à segurança, independente do seu domínio de aplicação, sejam processos industriais, máquinas, transportes ou energia. A IEC 61508 é amplamente aceita como a melhor norma genérica para gerenciamento de segurança funcional [7].

A norma define um sistema seguro como “livre de riscos inaceitáveis, envolvendo prejuízos físicos ou danos à saúde de pessoas, resultantes direta ou indiretamente de danos à propriedade ou ao ambiente” [7]. Ela apresenta uma abordagem genérica para todas as fases e atividades do ciclo de vida e um conjunto de técnicas e métodos para o desenvolvimento de equipamentos usados para realizar funções de segurança. A norma inclui também os procedimentos técnicos e administrativos necessários para atingir a integridade de segurança funcional desejada.

A norma descreve requisitos específicos para o desenvolvimento do hardware e do software para equipamentos eletrônicos programáveis [8], o que representa uma inovação em relação a normas anteriores, que inibiam o uso de sistemas programáveis devido aos riscos inerentes relacionados ao software.

2.2 Integridade de Segurança

Integridade de segurança é um conceito associado à probabilidade que a função de segurança seja executada satisfatoriamente e é indicada por um nível discreto chamado SIL. Os níveis previstos para SIL são 1, 2, 3 e 4, sendo que SIL 4 representa a maior integridade de segurança prevista pela norma. As exigências de projeto crescem do nível 1 ao 4.

Para hardware, a norma relaciona o SIL à taxa de defeitos perigosos resultantes de falhas não cobertas pelos mecanismos de detecção empregados e a restrições na arquitetura. O cálculo é realizado dispondo da taxa de defeitos de todos os componentes de hardware usados, da árvore de propagação de falhas, da distinção entre falhas

seguras e perigosas, do reconhecimento de falhas de modo comum, do grau de redundância da arquitetura e da cobertura dos mecanismos de detecção de falhas empregado, além de várias outras informações técnicas [8]. Se algum dos componentes é programável, a determinação do SIL alcançável deve incluir também a análise e verificação detalhada do software que executa no componente.

Todas as técnicas e estratégias usadas para o desenvolvimento e teste do hardware e do software do equipamento também são analisadas para determinar se um projeto tem condições de alcançar o SIL almejado. Essa análise visa determinar a robustez do equipamento a falhas sistemáticas, que são típicas de software e de projeto. Falhas sistemáticas diferem de falhas randômicas, que são típicas dos componentes de hardware. Para essa análise, a norma apresenta técnicas e medidas associadas às fases do ciclo de vida do equipamento, que vão desde a especificação até a retirada de operação e descarte [9]. Técnicas e medidas são classificadas como não-recomendadas (NR), recomendadas (R), altamente recomendadas (HR) ou não há recomendações contra ou a favor (--). A norma apresenta técnicas alternativas ou equivalentes, que devem ser escolhidas de acordo com sua adequação à aplicação. Para cada nível de SIL almejado, técnicas específicas são sugeridas ou impostas.

2.3 Certificação

É comum órgãos reguladores exigirem que equipamentos que executam funções de segurança sejam certificados segundo normas apropriadas. O objetivo de uma certificação de segurança é prover a garantia, reconhecida pelo órgão regulador, que o sistema foi considerado seguro pelo órgão certificador [7]. O conjunto de técnicas e medidas adotadas em todas as fases do ciclo de vida deve convencer o órgão certificador que o projeto apresenta o nível de integridade de segurança solicitado. Para o hardware são usados valores numéricos como taxa de defeito dos componentes e cobertura das técnicas de detecção e diagnóstico. No caso do software, não são usados valores numéricos: uma implementação deve seguir rigorosamente todas as recomendações da norma para o SIL desejado [11]. Considerando-se o SIL do hardware e do software, o menor deles será o SIL do sistema.

Quando uma recomendação não é seguida, é contrariada ou quando se usam alternativas não previstas, a certificação ainda é possível, mas a norma exige uma forte fundamentação baseada em argumentos técnicos [10]. Essa argumentação pode colocar um fator de incerteza na certificação, pois não é possível saber a priori se a argumentação será aceita pelo órgão certificador [11].

A função de segurança é certificada, principalmente, através da documentação gerada durante todas as fases do ciclo de vida do seu desenvolvimento. Cuidados devem ser tomados para definir documentos apropriados à certificação, mas a norma não estabelece o formato dos documentos. A certificação não é restrita ao produto gerado pelo desenvolvimento. Como a norma se aplica ao ciclo de vida, todo o processo de desenvolvimento e produção é levado em consideração para a certificação de segurança.

3 Metodologia de Desenvolvimento de Sistemas Seguros

O desenvolvimento dos equipamentos relacionados à segurança deve garantir a máxima cobertura de detecção de falhas randômicas possível para o SIL desejado. Deve também incluir medidas de prevenção de falhas. Todo o processo de desenvolvimento deve

seguir uma metodologia que inicia na especificação de segurança do sistema e contempla todas as etapas de implementação, verificação e validação. Testes devem acompanhar todas as etapas, da especificação à operação do equipamento em campo. A norma enfatiza a necessidade de planejar, especificar, desenvolver, testar e documentar os aspectos relativos à segurança funcional. A norma também sugere e, em alguns casos, impõe como essas atividades devem ser realizadas para alcançar um sistema com determinado nível de integridade de segurança.

A IEC 61508 é extensa, com 7 volumes, inúmeros anexos e centenas de páginas, textual na sua maior parte, e não convenientemente estruturada [7]. É fácil para um desenvolvedor se confundir com as inúmeras definições, técnicas e medidas impostas ou sugeridas, fases de projeto e documentação necessária. Neste contexto, deve ser considerado o emprego de uma ferramenta que incorpore, de forma estruturada, o conhecimento necessário para a aplicação da norma.

3.1 Tolerância a Falhas e a Norma IEC 61508

O desenvolvimento de hardware e software deve incorporar métodos e técnicas da área de tolerância a falhas. Para isso, recorre-se ao emprego de redundância de componentes, diagnóstico e detecção de falhas, diversidade de componentes de hardware e de software, degradação suave e à aplicação de uma série de proteções extras, de maneira a atingir um estado seguro no caso de ocorrência de falhas. Entretanto, nem todas as técnicas usuais de tolerância a falhas são recomendadas, e algumas são explicitamente não-recomendadas. Por exemplo, a técnica popular de recuperação para um estado anterior da computação não é recomendada para SIL 4. Recuperação para um novo estado posterior a falha, que aparecia como técnica recomendada em 2000, não é mais citada na edição de 2010. Correção de falhas usando inteligência artificial não é recomendada de SIL 2 a 4, sendo indiferente para SIL 1. Assim não basta o bom senso e a experiência do desenvolvedor: a norma deve ser conferida a cada escolha de técnica ou medida a ser empregada.

A norma assume que falhas de hardware e software são inevitáveis. O erro causado pela falha deve ser detectado e o sistema deve corrigir o erro ou ir para um estado seguro. Mas é impossível detectar todos os erros. A taxa residual de defeitos, devido à cobertura imperfeita dos mecanismos de tolerância a falhas, detecção e diagnóstico empregados irá, em última análise, determinar o nível de integridade de segurança do equipamento; quanto menor essa taxa, maior o SIL.

Sistemas com menor probabilidade de apresentar defeitos residuais da função de segurança são classificados em SIL 4 (10^{-9} defeitos perigosos por hora ou 10^{-5} defeitos sob demanda em baixa demanda de operação). Aumentando uma ordem de grandeza na taxa de defeitos, diminui de uma unidade o valor do SIL, até SIL 1. Por exemplo, uma função de segurança de detecção de incêndio que apresente, no máximo, um defeito a cada 10.000 demandas, seria classificada como SIL 3. Um defeito no caso corresponde a não atuar em caso de incêndio.

Vale notar que a norma não altera ou inova em relação a boas práticas de engenharia de software e ao emprego de técnicas de tolerância a falhas. Mas exige comprovação documentada da eficiência de seu emprego em um dado projeto.

3.2 Desenvolvimento de Software Seguro

Boas práticas de projeto não são suficientes para garantir a segurança necessária [6] para sistemas de segurança. Ao contrário dos procedimentos aplicados ao hardware [11], não há mecanismos universalmente aceitos para determinação da taxa de defeitos de software [11]. Na IEC 61508, a ênfase está na prevenção e controle de falhas. Deve ser cuidadosamente controlado o modo como as atividades de desenvolvimento de software são realizadas. Devem ser enfatizadas boas práticas, técnicas e medidas que auxiliem alcançar a integridade de segurança desejada.

Um ciclo de vida voltado à segurança deve incluir procedimentos técnicos e administrativos. Entre os técnicos estão a integração de técnicas e medidas voltadas à segurança e as atividades do ciclo de vida do equipamento. Entre os procedimentos administrativos estão o planejamento da segurança funcional e procedimentos rígidos para a gerência de configuração de software. Os procedimentos administrativos estão relacionados à gerência de qualidade do software.

O planejamento de segurança define a estratégia para a obtenção, o desenvolvimento, a integração, a verificação, a validação e a modificação do software de acordo com o nível da integridade de segurança (SIL) do sistema. Uma função importante da gerência da qualidade é especificar as responsabilidades das pessoas, departamentos e organizações em cada uma das atividades do ciclo de vida de projeto do software [3]. A gerência de qualidade deve também definir procedimentos para acompanhar o sistema em operação e avaliar se a taxa de defeitos está de acordo com o assumido. Para permitir auditoria, além dos documentos relacionados ao projeto todas as atividades de gerência devem também ser documentadas [9].

3.3 Ciclo de Vida para Segurança de Software

Uma das funções da gerência de qualidade de software é definir um ciclo de vida estruturado em fases e atividades. A norma não impõe um ciclo de vida. É enfatizado, entretanto, que procedimentos para garantir qualidade e segurança compatíveis com aqueles contemplados na norma devem ser incorporados nas atividades do ciclo de vida. Cada fase do ciclo de vida deve ser dividida em atividades elementares. Para cada atividade elementar devem ser definidos o seu escopo, as entradas e as saídas. Além disso, cada atividade deve ser documentada.

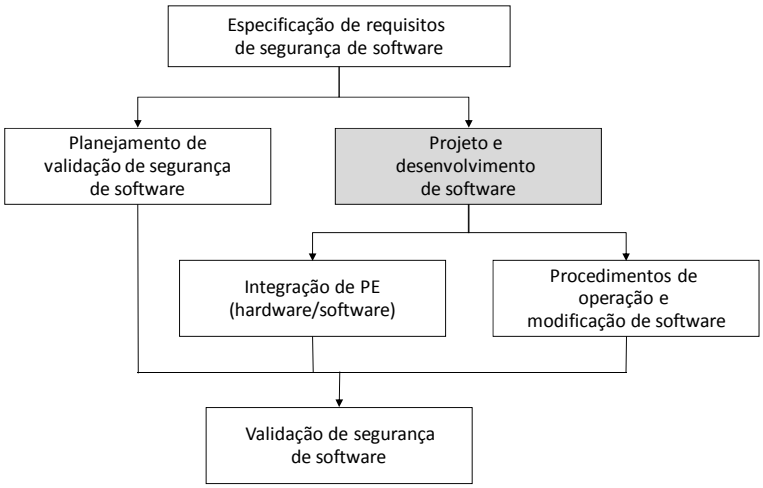

Fig. 1 Ciclo de vida de software seguro

Apesar de não impor, a norma sugere um modelo para ciclo de vida para software (figura 1).

Os requisitos para as funções de segurança e os requisitos para a integridade de segurança são diferentes: os primeiros são derivados da análise de ameaças e indicam o que a função de segurança efetivamente realiza. A integridade de segurança, por outro lado, é a probabilidade que a função de segurança seja executada satisfatoriamente, e os requisitos são derivados da estimativa de riscos. Deve ser enfatizado que as atividades da figura 1 podem fazer parte do ciclo de vida de qualquer bom projeto de software, mesmo aqueles sem necessidade de certificação.

3.4 Projeto e Desenvolvimento de Software

O modelo V (fig 2) definido na IEC 61508 mostra a fase de projeto e desenvolvimento de software do ciclo de vida . O modelo V inicia com a especificação dos requisitos de segurança de software, que é derivada da especificação dos requisitos de segurança global do sistema.

A arquitetura de software é definida a partir da especificação, mas é também influenciada pela arquitetura de hardware do equipamento de segurança. A fase de definição da arquitetura de software estabelece as estratégias de tolerância a falhas e diagnóstico que serão implementadas em software. A fase envolve também a escolha das ferramentas de suporte e linguagens de programação adequadas.

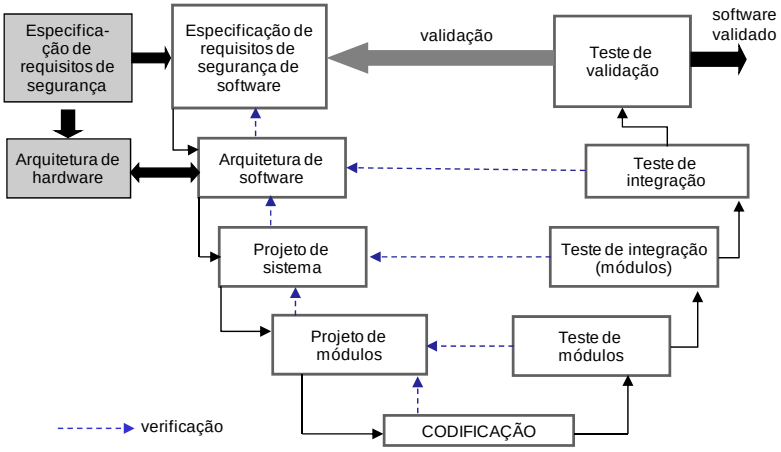

Fig. 2 Modelo V

Validação, verificação e avaliação envolvem técnicas e resultados diferentes. De acordo com a norma, a validação da segurança funcional deve garantir que a integração dos componentes de software e hardware do sistema atenda aos requisitos especificados para a segurança do software no SIL requerido. A verificação de software em relação ao SIL desejado deve testar e avaliar as saídas de uma dada fase do ciclo de vida para garantir correção e consistência em relação às entradas da fase. Finalmente a avaliação da segurança do software deve investigar e julgar a integridade de segurança funcional alcançada pelo sistema.

Para cada uma das atividades do modelo V, a norma associa tabelas que devem ser seguidas. Nessas tabelas constam as técnicas recomendadas para cada nível de integridade de segurança e a documentação que corresponde à atividade. O desenvolvedor deve assinalar nas tabelas todas as técnicas empregadas assim como

justificativas se uma dada técnica recomendada não foi aplicada. A apresentação das tabelas para o desenvolvedor, com a definição dos termos e conceitos, referência à sua localização na norma e espaço para armazenamento e recuperação de documentos, forma a base para ferramentas assistentes de projeto nesta área.

4 Ferramentas de Projeto Visando SIL

Várias ferramentas para auxiliar a aplicação da IEC 61508 estão disponíveis no mercado. Geralmente permitem o cálculo de SIL do hardware e suporte a geração e armazenamento de documentos. Três das mais completas ferramentas são Silcore, exSILentia , e SilSover. Silcore [13] é uma ferramenta da ACM Automation Inc para o projeto e avaliação de sistemas desenvolvidos visando cumprir os requisitos das normas IEC 61508IEC, IEC 61511, e ANSI / ISA 84.0, e é baseada no ciclo de vida do sistema. A ferramenta facilita otimizar o SIL, através do cálculo do intervalo de teste apropriado para o sistema em operação, e dos cálculos de confiabilidade necessários à determinação do SIL do hardware. Para as fases de manutenção e operação, ela oferece opções de rastreamento e geração de relatórios para equipamentos fora de serviço.

ExSILentia [14], da Exida, conta com um módulo para a geração automática de documentação, ficando a cargo do usuário preencher detalhes específicos do projeto. Outros módulos da ferramenta auxiliam na análise de risco e seleção do SIL e ajudam na especificação dos requisitos de segurança. O módulo SILver é a parte responsável pela verificação do SIL, e possui motor de cálculos para SIL de hardware certificado por terceiros. Os dados de confiabilidade dos componentes de hardware podem ser obtidos de uma base de dados. Para as fases finais do ciclo de vida, a ferramenta oferece geração automática de procedimentos para testes e gravação de dados de eventos como testes, falhas e picos de demanda.

SilSolver [15] permite que sejam indicados os dispositivos que compõem o sistema sendo avaliado e que se descreva sua topologia. A partir dessa descrição, a ferramenta avalia o SIL dos componentes de hardware aplicando uma árvore de falhas e informando ao usuário dados como as taxas de defeito do equipamento. A ferramenta permite a visualização da contribuição dos diversos dispositivos para o resultado final da análise, e também fornece apoio à documentação do projeto através de relatórios.

A ferramenta desenvolvida neste trabalho, SVA (Software Validation Assistent), foca no desenvolvimento do software crítico e não no hardware como as demais citadas acima. A principal vantagem da ferramenta em relação às demais é a sua adaptabilidade e facilidade de extensão para outras funcionalidades, inclusive para adaptações a alterações futuras na norma. Deve-se enfatizar que a ferramenta proposta não visa atuar sobre todo o ciclo de vida do sistema, mas apenas nas fases relacionadas ao desenvolvimento de software.

5 SVA: Software Validation Assistant

O objetivo do assistente de validação de software, SVA, cujo protótipo é apresentado neste artigo, é servir como ferramenta no apoio ao projeto de software a ser desenvolvido em conformidade com a norma IEC 61508.

Um assistente é útil para o desenvolvimento de software seguro, pois as técnicas e medidas sugeridas ou impostas pela norma nem sempre correspondem à experiência usual dos desenvolvedores, o que aumenta a complexidade na sua aplicação. As

ferramentas disponíveis no mercado são de alto custo, o que inibe seu uso para aprendizagem e familiarização. Para os desenvolvedores adquirirem habilidades na aplicação da norma, e também para fins de treinamento e aprendizagem, uma ferramenta como o SVA pode ser de grande auxílio.

A garantia de certificação não está assegurada com o uso do SVA ou de qualquer outra ferramenta comercial, mesmo seguindo todos os passos corretamente, uma vez que não existem meios algorítmicos para avaliar determinada documentação. Entretanto, uma ferramenta de apoio guia a equipe de desenvolvimento através do ciclo de vida informando se todas as recomendações e atividades previstas no ciclo foram cumpridas. O cumprimento de todas as exigências e a estruturação facilitada pela ferramenta facilita o processo de certificação.

O SVA é uma ferramenta de apoio que indica para cada fase do ciclo de vida quais as técnicas que devem ser aplicadas, quais os documentos e artefatos devem ser gerados na fase e quem são os responsáveis pela fase. Nesse sentido ela é um guia para a aplicação da norma.

5.1 Implementação do Assistente de Validação

O assistente de validação, SVA, executa em um servidor Apache Tomcat, sendo suas páginas desenvolvidas em JSP (Java Server Pages). Apache Tomcat é um servidor web Java, ou mais especificamente um container de servlets. JSP é uma tecnologia para aplicações web semelhante a ASP e PHP. JSP permite ao desenvolvedor produzir aplicações que acessem o banco de dados, manipulem arquivos no formato texto, capturem informações a partir de formulários e captem informações sobre o visitante e sobre o servidor. O sistema SVA usa uma instância de banco de dados relacional Apache Derby para efetuar o armazenamento e manipulação dos dados.

5.2 Classes do Assistente de Validação

O diagrama de classes é uma representação da estrutura e relações das classes que servem de modelo para objetos e serve como base para a construção de outros elementos da documentação relativa à ferramenta. O sistema SVA (figuras 3 e 4) divide-se entre 11 classes e três enumerações, totalizando 14 classes.

Foram definidos três tipos de usuários para o SVA: o gerente que pode criar novos projetos e designar desenvolvedores para as diferentes fases do ciclo de vida; o especialista na norma, chamado técnico, que pode atualizar o sistema, e os desenvolvedores que implementam as funções de segurança seguindo passo a passo as recomendações da norma.

A classe *Usuario* (fig. 4) representa a abstração dos dados de usuários (gerente, técnico ou desenvolvedor), com os atributos email, nome, perfil e username. *Projeto* (fig. 3) representa os objetos que contém nomeProjeto, nomeEmpresa, silPretendido, gerenteProjeto e responsavelTecnico (sendo os 2 últimos instâncias da classe *Usuario*).

O SIL pretendido irá definir as técnicas e métodos recomendados e não recomentados para um dado projeto em todas as faces do ciclo de vida. O SIL faz parte da especificação da função de segurança e não pode ser alterado. Ao desenvolvedor serão mostradas apenas as informações necessárias para alcançar o SIL pretendido.

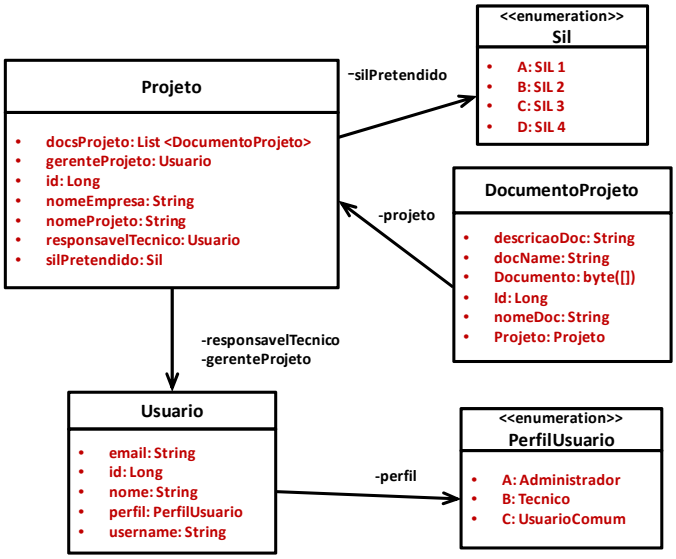

Fig. 3 Diagrama de classes: projeto e usuário

CicloVida (fig. 4) define os atributos dos objetos que representam as fases do ciclo de vida, tais como código, nome, descrição e uma lista de técnicas que são referenciadas. Ao ciclo de vida e ao SIL pretendido estão associadas às técnicas e medidas sugeridas ou impostas pela norma.

A norma organiza as técnicas em tabelas de dois níveis. Várias técnicas gerais do primeiro nível são detalhadas em tabelas do segundo nível. Por exemplo, na fase de especificação dos requisitos de segurança de software (fig 2), a tabela de primeiro nível indica que o uso de métodos semi-formais de especificação é altamente recomendado para SIL 3. A tabela de segundo nível lista os métodos (diagramas funcionais, diagramas de sequência, máquinas de estado finitas, redes de Petri, tabelas de decisão, ...) e seu grau de recomendação. Os dois tipos de tabela são representados por *TecnicaMedidaA* e *TecnicaMedidaB*.

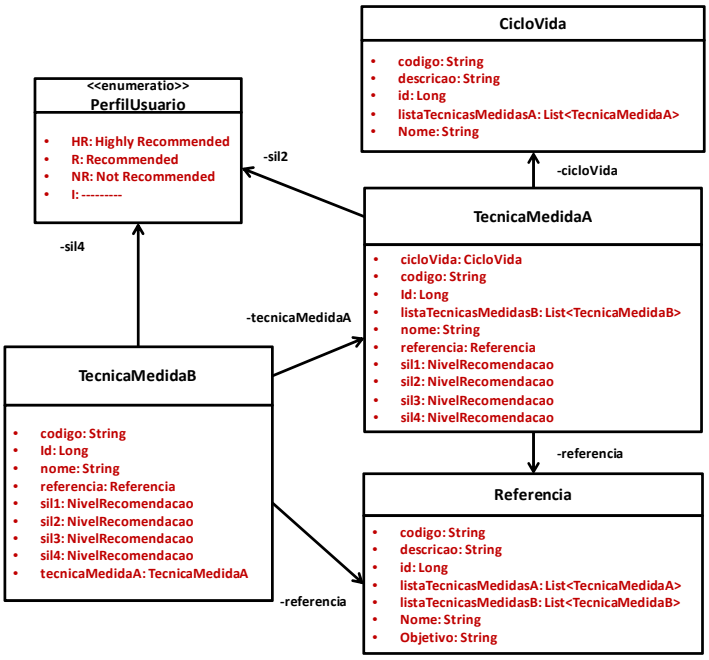

Fig. 4 Diagrama de classes: ciclo de vida

TecnicaMedidaA representa técnicas e medidas através dos atributos codigo, nome, cicloVida (referência a qual fase do ciclo de vida a técnica pertence), sil1, sil2, sil3, sil4, além de uma lista de técnicas das tabelas da norma as quais referencia. *TecnicaMedidaB* por ser um detalhamento das técnicas gerais, liga-se um objeto do tipo *TecnicaMedidaA*. Ambas estão relacionadas à classe *Referencia*, que possui os atributos codigo, nome, descricao e objetivo, além de listas de técnicas (das tabelas dos anexos A ou B da IEC 61503, parte 3) pelas quais são referenciadas.

As classes listadas a seguir completam o assistente e auxiliam na gerência e documentação de um projeto:

- *EventoAgenda*: representa eventos da agenda, com os atributos data, hora, nomeEvento e infoEvento;
- *DocumentoConhecimento*: abstrai objetos que contém o arquivo, o nome dado pelo usuário (nomeDoc), a descrição do arquivo e o nome do arquivo no sistema (docName);
- *LinkConteudo*: representa objetos formados pelo nome, assunto e endereço url;
- *Topico*: abstrai objetos que contém nomeTopico, assuntoTopico e conteudoTopico, no formato String;
- *DocumentoProjeto*: semelhante a *DocumentoConhecimento*, com a adição de um atributo que faz referência a um projeto.
- As enumerações são: *NivelRecomendacao*, *PerfilUsuario*, *Sil*. Seus valores representam um conjunto finito de identificadores previamente definidos.

5.3 Interface com Usuários

A seguir são mostradas algumas das funcionalidades do SVA acessível através da interface com os usuários do sistema. A tela de cadastro de projetos (figura 5) permite cadastrar um projeto com o seu nome, a empresa para a qual está sendo desenvolvido, o SIL pretendido, além do gerente e responsável técnico. Os dois últimos são preenchidos através de uma janela *popup* de busca de usuários já cadastrados, que é mostrada quando do clique no botão "..." do campo desejado.

Fig. 5 Cadastro de projetos

A figura 6 mostra os resultados de uma busca de usuários. O usuário desenvolvedor alocado a um projeto com dada responsabilidade pode agora iniciar o processo de acompanhamento e validação da sua tarefa.

Ao clicar no menu Validação, são apresentados 4 submenus (SIL 1, SIL 2, SIL 3 e SIL 4). Ao selecionar o SIL desejado, o sistema carrega uma tela com o ciclo de vida e as técnicas a ele relacionadas, bem como o índice de referência da técnica e o nível de recomendação para o SIL selecionado.

Fig. 6 Seleção de usuários

As informações sobre as fases do ciclo de vida de segurança de software, bem como suas respectivas técnicas, encontram-se nas tabelas A.1 até A.10 do Anexo A da parte 3 (volume 3) da IEC 61508. O código do ciclo de vida representa a qual tabela de técnicas da norma ele está associado. No caso de uma técnica, representa o próprio código da técnica dentro da tabela da fase do ciclo de vida a qual pertence.

Para efetuar a validação, o usuário seleciona as técnicas que estão sendo utilizadas através da *checkbox* associada a cada técnica. Ao acionar o botão Validar, o sistema verifica se alguma técnica HR (altamente recomendada) não foi marcada, e/ou se alguma técnica NR (não recomendada) foi selecionada. Caso aconteça, o sistema avisa que em ambos os casos há a necessidade de apresentar uma justificativa detalhada que será julgada pelo órgão certificador durante o processo de certificação. A figura 7 mostra a tela com um exemplo de validação para SIL 1.

Fig. 7 Exemplo de tela de validação para SIL 1

Finalmente, no menu Área Técnica estão presentes as funcionalidades que permitem o gerenciamento do ciclo de vida e do modelo V, bem como das técnicas e medidas presentes nas tabelas dos anexos A e B da IEC 61508 parte 3, além das respectivas referências. A área técnica tem por objetivo proporcionar flexibilidade à ferramenta, para torná-la adaptável a eventuais mudanças, como por exemplo, uma nova edição da norma.

5.4 Avaliação da Ferramenta SVA

A ferramenta está sendo utilizada pelos bolsistas e pesquisadores do projeto RIO-SIL, durante as atividades de treinamento do projeto. Esses usuários estão avaliando a ferramenta quando utilizada sob os três papeis que ela disponibiliza: gerente, técnico e desenvolvedor. Algumas funcionalidades, como rastreamento de mudança e geração de

relatórios, ainda estão sendo implementadas. Até o momento, a ferramenta vem cumprindo com seus objetivos de facilitar a compreensão e aplicação da norma IEC 61508 durante treinamento de equipe. Uma avaliação mais completa será possível quando a ferramenta for utilizada não apenas para treinamento, mas para a implementação de um projeto real de uma função de segurança.

6 Conclusão

A IEC 61508 estabelece os requisitos necessários para assegurar que os sistemas sejam projetados, implementados e operados para suprir as exigências do nível de integridade de segurança (SIL) requerido. A norma define também um processo a ser seguido por todas as partes envolvidas, visando uniformizar a terminologia e facilitar a certificação. Embora elogiada por sua adaptabilidade a vários setores, a norma também recebe críticas. A IEC 61508 é deficiente por deixar muitos pontos vagos em relação à documentação e ao tratamento das justificativas quando uma técnica sugerida ou imposta pela norma não é aplicada. Muitas vezes também é necessário recorrer a outras normas para completar os procedimentos necessários para o processo de certificação.

No Brasil, a crescente demanda por equipamentos certificados para segurança funcional faz com o desenvolvimento de software seguro seja igualmente impulsionado. A ferramenta SVA tem o objetivo de auxiliar o desenvolvedor a aplicar técnicas e medidas relacionadas à norma e a manter o registro do caminho percorrido de acordo com o ciclo de vida determinado para o projeto. Tem por objetivo servir como uma alternativa de ferramenta de apoio à aplicação de normas para sistemas relacionadas à segurança, uma vez que as ferramentas existentes não são facilmente disponíveis.

O SVA abrange todas as fases do ciclo de vida de segurança de software, permitindo a inclusão de novas fases e a alteração das fases existentes. Isto é útil para tornar o sistema adaptável a eventuais mudanças na norma, mesmo que estas não ocorram com grande frequência. A mesma facilidade de extensão e alteração é apresentada para as técnicas relacionadas às fases do ciclo de vida, o que proporciona flexibilidade à aplicação. Além de auxiliar na verificação do emprego das técnicas pertinentes, o SVA permite a manutenção de registros de usuários, projetos e eventos. Ela oferece apoio também para a criação de uma base de conhecimento, através de funcionalidades de cadastro e busca de links, tópicos e documentos. Isto possibilita a criação de um banco de informações que sejam relevantes ao contexto e que estejam disponíveis para consulta e acesso rápido pelos usuários do sistema.

No treinamento de desenvolvedores no uso na norma IEC 61508, a prática mostra que a apresentação puramente textual da norma é cansativa [16], o que pode levar a sua aplicação incorreta. Sem o auxílio de uma ferramenta, o tempo de assimilação de todos os detalhes envolvidos na norma IEC, nos seus sete volumes e centenas de páginas é desencorajador. A ferramenta mantém a atenção a cada detalhe do ciclo de vida no momento onde a informação é necessária tornando o desenvolvimento mais produtivo e menos sujeito a erros por omissão ou má interpretação.

7 Referências

[1] Avizienis, A.; Laprie, J.-C.; Randell, B.; Landwehr, C., “Basic concepts and taxonomy of dependable and secure computing.” *Dependable and Secure Computing, IEEE Transactions on* , vol.1, no.1, pp. 11- 33, Jan.-March 2004

[2] Dunn, W. R., “Designing safety-critical computer systems.” *IEEE Comp*, 36(11):40 – 46. 2003.

[3] Bell, R., “Introduction and Revision of IEC 61508”. *Advances in Systems Safety*, 2011,Springer

[4] Faller, R., “Project experience with IEC 61508 and its consequences.” SAFECOMP 2001, v. 2187 of Lecture Notes in Computer Science, pp 200 – 214.

[5] *International Electrotechnical Commission IEC 61508, part 1 to 7; Functional Safety of Electrical, Electronic and Programmable Electronic Safety-Related Systems*. IEC Std. 2010. http://www.iec.ch/functionalsafety.

[6] Johnson, C., “Using IEC 61508 to guide the investigation of computer-related incidents and accidents”. In SAFECOMP 2003, v. 2788 of Lecture Notes in Computer Science, pages 410 – 424.

[7] Panesar-Walawege, R.K. et all. “Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard”, in *2010 Third International Conference on Software Testing, Verification and Validation*. 335-344

[8] Smith D.J.; Simpson, K.G.L.; *Functional Safety: a straightforward guide to applying IEC 61508 and related standards*, Elsevier, Butterworth-Heinemann, U.K. 2ªedição, 2004.

[9] Brown, S. “Overview of IEC 61508 Design of electrical/electronic/programmable electronic safety-related systems”. *IEEE Computing and Control, Engineering Journal*, fev. 2000.

[10] Mcdermid, J.A. “Software Safety: Where’s the Evidence?”, in *Proc. 6th Australian Workshop on Industrial Experience*, v.3, 2001.

[11] Cechin, Sergio Luis; Weber, Taisy Silva; Netto, Joao Cesar. Arquiteturas MooN(D) para portas de entrada e saída de remotas em conformidade com a IEC 61508 . In: *Congresso Brasileiro de Automática* (19. : 2012 set. 02-06 : Campina Grande, PB). Campinas, SP : Sociedade Brasileira de Automática, 2012. p. 4500-4507.

[12] Mayr, A.; Plösch, R.; Saft, M.; , "Towards an Operational Safety Standard for Software: Modelling IEC 61508 Part 3," *Engineering of Computer Based Systems (ECBS), 2011 18th IEEE International Conference and Workshops on* , vol., no., pp.97-104, 27-29 April 2011

[13] (2013) Silcore. [Online]. Available: http://www.acm.ab.ca/

[14] (2013) exSILentia. [Online]. Available: http://www.exida.com/

[15] (2013) SilSolver. [Online]. Available http://www.sis-tech.com/

[16] Weber, Taisy Silva; Cechin, Sergio Luis; Netto, Joao Cesar. Integridade de segurança em sistemas críticos de controle e instrumentação . In: *Conferência Internacional em Tecnologias Naval e Offshore : ciência e inovação* (1. : 2012 março 22-23 : Rio Grande, RS), Rio Grande, RS : FURG, 2012. [4] f.