

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

LUKAS LAMPERT

**Ferramenta de Apoio ao Desenvolvimento de
Software Embarcado de Acordo com a
Norma IEC-61508 Ed 2.0**

Trabalho de Graduação.

Prof. Dra. Taisy Weber
Orientadora

Porto Alegre, Dezembro de 2012

UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL

Reitor: Prof. Carlos Alexandre Netto

Vice-Reitor: Prof. Rui Vicente Oppermann

Pró-Reitoria de Graduação: Prof.^a Valquíria Linck Bassani

Diretor do Instituto de Informática: Prof. Luís da Cunha Lamb

Coordenador da ECP: Prof. Sérgio Cechin

Bibliotecária-Chefe do Instituto de Informática: Beatriz Regina Bastos Haro

SUMÁRIO

| | |
|--|-----------|
| LISTA DE FIGURAS | 4 |
| RESUMO | 5 |
| ABSTRACT | 6 |
| 1 INTRODUÇÃO | 7 |
| 2 SEGURANÇA | 8 |
| 2.1 Histórico de Segurança | 8 |
| 2.2 Exemplos de Uso | 8 |
| 2.3 Definições de Segurança | 8 |
| 2.4 Certificação de Software | 9 |
| 3 NORMA IEC61508 | 11 |
| 3.1 IEC | 11 |
| 3.1.1 IEC61508 | 11 |
| 3.2 Objetivos | 12 |
| 3.3 Estrutura | 12 |
| 3.4 Edição revisada 2.0 de 2010 | 13 |
| 3.5 Considerações sobre a norma | 14 |
| 3.6 Alcançando Segurança Funcional | 15 |
| 3.7 Ciclos de Vida | 15 |
| 3.8 Ciclo de Vida de Segurança Funcional | 16 |
| 4 IEC61508-3 | 19 |
| 4.1 Segurança de Software | 19 |
| 4.2 Ciclo de vida de Segurança de Software | 19 |
| 4.3 CICLO DE PROJETO E DESENVOLVIMENTO DE SOFTWARE | 20 |
| 4.4 Artefatos | 22 |
| 4.5 Backtracking | 22 |
| 5 IMPLEMENTAÇÃO | 23 |
| 5.1 Funcionalidades | 23 |
| 5.2 Desenvolvimento da Ferramenta | 23 |
| 5.2.1 Arquitetura | 23 |
| 5.2.2 Ferramentas utilizadas | 25 |
| 5.2.3 Bibliotecas utilizadas | 25 |
| 5.2.4 Detalhes do Desenvolvimento | 25 |
| 6 CONCLUSÃO | 31 |
| BIBLIOGRAFIA | 32 |
| ANEXO <ARTIGO TG1: DESENVOLVIMENTO DE SOFTWARE EMBARCADO DE ACORDO COM A NORMA IEC-61508 ED 2.0 (2010)> | 34 |

LISTA DE FIGURAS

| | |
|---|----|
| Figura 3.1: Origem das causas primárias de falhas em sistemas de controle. | 17 |
| Figura 3.2: Ciclo de vida de Segurança Funcional. | 18 |
| Figura 4.1: Ciclo de vida de Segurança de Software | 21 |
| Figura 4.2: Modelo V de desenvolvimento de software | 23 |
| Figura 5.1: Tela inicial do SSCSideKick!, com o menu de arquivo selecionado. | 27 |
| Figura 5.2: Interface de criação de projeto. | 27 |
| Figura 5.3: Arquivo de configuração de norma no formato XML. | 28 |
| Figura 5.4: Janela para edição de um projeto. | 28 |
| Figura 5.5: Modelo do arquivo de saída de um projeto no formato XML..... | 29 |
| Figura 5.6: Interface para criação de usuário. | 30 |
| Figura 5.7: Visualização de técnicas da atual fase..... | 30 |
| Figura 5.8: Interface para alteração de regra. | 31 |
| Figura 5.8: Verificação de SIL..... | 31 |
| Figura 5.9: Arquivo de log..... | 32 |

RESUMO

A utilização de sistemas eletrônicos é cada vez mais comum nos dias de hoje, estando presentes nas mais diversas áreas. De grande benefício à sociedade, esses sistemas muitas vezes também tem o poder de prejudicar quando neles ocorrem falhas: sejam danos a seres vivos, ambientais ou até econômicos. Quando se trata de sistemas críticos, deseja-se ter a garantia que esses sistemas sejam suficientemente seguros. Na busca de certificações de segurança para sistemas eletrônicos, destaca-se a norma IEC61508.

No decorrer deste trabalho são vistos conceitos relacionados à segurança, e um estudo da norma IEC61508 na sua versão mais atual, 2.0 de 2010. Será dado maior enfoque na terceira parte da norma, referente ao desenvolvimento de software seguro. Ao final, é apresentada uma ferramenta digital para o auxílio aos projetistas de software no complexo processo de certificação, tendo como base os Requisitos de Software da IEC61508-3.

Palavras-chave: IEC 61508, norma, requisitos, risco, segurança, software.

IEC61508 Ed 2.0 (2010) Compliance Embebed Software Development

ABSTRACT

The utilization of electronic systems is very common nowadays; being present in a wide range of areas. Those systems provides society with great benefit, however are capable of causing harm in the event of failure: may it be on living beings, environment or economics. When dealing with critic systems, people want to have the warranty that those systems are sufficiently safe. On the quest for certifications for electronic systems, IEC61508 standard is noticed.

This work reviews general safety related concepts, and provides a study of the most recent version of the IEC61508 standard: 2.0 from the year 2010. More attention is given to the third part of the standard, which refers to software safety requirements. At the end a digital tool is presented in order to help software developing teams in the complex process of obtaining software safety certification, based on the Software Requirements from IEC61508-3.

1. INTRODUÇÃO

Times de desenvolvimento de sistemas eletrônicos buscam nas certificações de segurança a garantia de que seus produtos tenham o menor impacto negativo possível quando na presença de falhas. O padrão IEC61508 é de notável importância nesse contexto. Há hoje um projeto no Instituto de Informática da UFRGS cujo objetivo é o desenvolvimento de sistemas seguros relacionados à indústria de óleo e gás, em conformidade com esta norma [12] [13]. Buscando artifícios que auxiliassem o desenvolvimento desses sistemas, percebeu-se a ausência de ferramentas que suprissem algumas necessidades na parte de desenvolvimento de software e que fossem gratuitas.

Este trabalho tem por objetivo a obtenção de conhecimento a respeito da norma IEC61508, principalmente sobre a parte de Requisitos de Software, além de oferecer uma solução digital que auxilie as equipes de desenvolvimento na difícil tarefa de se buscar uma certificação internacional de segurança.

O escopo deste trabalho se limita a parte de requisitos de software, que é apresentado em IEC61508-3. É importante levar em consideração que a utilização da ferramenta não garante a aceitação de um projeto por uma entidade certificadora, ela apenas visa facilitar o processo.

Apesar de a ferramenta ser focada na norma IEC61508-3 de 2010, ela é facilmente adaptável a futuras versões da norma, bem como outras certificações que sigam o mesmo contexto: desenvolvimento dividido em fases; cada fase contendo uma série de regras ou técnicas, sendo elas classificadas em diferentes níveis de integridade de segurança.

Mesmo havendo grande possibilidade de melhoria com a adição de inúmeras funcionalidades, a ferramenta apresentada cumpre seu papel em auxiliar no desenvolvimento de sistemas seguros, com alto nível de adaptabilidade.

2. SEGURANÇA

Este capítulo oferece um entendimento dos conceitos básicos de segurança no contexto de sistemas eletrônicos. Veremos um histórico do assunto e alguns exemplos de onde é aplicado. Em seguida entramos em detalhe na certificação de software.

2.1. Histórico de Segurança

A partir da década de 80, houve o crescimento na utilização de sistemas relacionados à segurança, em campos onde a garantia de funcionamento é um fator crítico, como a aviação, usinas nucleares, equipamentos médicos, ferrovias, aplicações militares e indústria. Juntamente a essa ascensão, houve uma série de iniciativas ao redor do mundo na criação de padrões e guias que permitissem o desenvolvimento desses sistemas, para que fossem comprovadamente seguros. Dentro dessas iniciativas, uma de grande destaque é a norma IEC61508, que é abordada no presente trabalho. [3], [6], [4]

2.2. Exemplos de Uso

Podemos ilustrar alguns sistemas onde a segurança é necessária, e a busca por certificações é vista como essencial:

- Em automóveis: nos sistemas de tração e freios ABS;
- Em maquinários e equipamentos na indústria ou indústrias de processos químicos danosos: sistemas de parada emergencial;
- Em hospitais: aparelhos de suporte a vida;
- Equipamentos utilizados em radioterapia: mecanismos de controle de dosagem de exposição e bloqueio;
- Em aeronaves: sistemas de controle de voo e orientadores de mísseis;
- Em ferrovias: mecanismos de sinalização;
- Em usinas nucleares: sistemas de desligamento emergencial [4]

2.3. Definições de Segurança

A fim de proporcionar uma melhor compreensão do presente trabalho, iremos rever a definição de alguns termos relacionados à segurança. Essa terminologia também é apresentada em língua inglesa.

- Hazard: qualquer condição que possa causar danos, ferimentos, ou morte à seres vivos; dano ou perda de sistemas, equipamentos ou propriedade; ou danos ao ambiente. Um elemento que possua o potencial para causar um *hazard* necessitará de um *mecanismo de iniciação* para ocorrer. Este por sua vez é a sequência de eventos que leva ao *hazard* acontecer. *Hazards* são a base para a definição de requisitos de segurança. [7]
- Risco (Risk): no processo de desenvolvimento de um Sistema Relacionado à Segurança, cada *hazard* é avaliado em termos do risco que ele oferece. Risco pode ser definido como a combinação da probabilidade de ocorrer determinado dano e a severidade do mesmo, caso ocorra. A possibilidade pode ser classificada qualitativamente em: frequente, provável, ocasional, remota, improvável e incrível. O

nível de dano causado também pode ser classificado como: catastrófico, crítico, marginal ou negligenciável. [7]

- Segurança (Safety): pode ser definido com a ausência de risco inaceitável de gerar um *hazard* [3], ou a segurança de funcionamento em situações críticas [8].
- Segurança Funcional (Functional Safety): é a capacidade de um sistema ou equipamento operar corretamente de acordo com as suas entradas. Deve-se considerar o sistema como um todo, inclusive sua interação com o ambiente [3].
- Função de Segurança (Safety Function): é a função implementada em um Sistema Relacionado à Segurança que é responsável por alcançar ou manter um estado seguro do equipamento no evento de um potencial *hazard* específico [3]. No contexto da norma IEC 61508, cada Função de Segurança é definida com um grau de integridade: o *safety integrity level (SIL)* [7].
- Safety Integrity Level (SIL): O Nível de Integridade de Segurança determina o grau de desempenho de segurança. Na norma em questão, varia em 4 níveis, sendo SIL4 o mais exigente, e SIL1 o de menor grau [3].
- Sistema Relacionado à Segurança (Safety-related System): é um sistema que possui uma ou mais Funções de Segurança e é capaz de utilizá-las quando necessário, de acordo com um Nível de Integridade de Segurança ao qual ele foi projetado, a fim de evitar o acontecimento de *hazards* previstos para esse sistema. [3]
- Software Criticality: a análise de *software criticality* permite diminuir o grau de exigência necessário para se alcançar um SIL, e, conseqüentemente diminui o esforço necessário para se demonstrar a segurança. No contexto da IEC 61508-3, a *criticality* possui 4 níveis (de C0 a C3) que, cruzadas com os 4 níveis de SIL impõe condições e exigências necessários para cada caso. Esta abordagem se limita, entretanto, a unidades de software com funcionalidades e interfaces restritas, como bibliotecas e *drivers* para sistemas operacionais. [6]
- Safety Case: é uma estrutura na qual a equipe de desenvolvimento faz declarações de como os requisitos de segurança foram atingidos no produto em questão, justificando as medidas e técnicas tomadas. A criação desses documentos que compõe o *safety case* podem complementar o processo de desenvolvimento de sistemas seguros. [7] Existem abordagens para certificação que levam em consideração somente este artefato, ao invés da avaliação de todas as etapas do projeto. Essas abordagens permitiriam mais liberdade ao desenvolvimento, mas não são bem vistas quando tratam de Sistemas Relacionados à Segurança, uma vez que gera uma grande dificuldade na avaliação pelas entidades certificadoras. [4]

2.4. Certificação de Software

Validação e verificação de qualidade de *hardware*, tanto para uso em segurança ou não, estão hoje no seu estado da arte. As medidas dos seus atributos são precisas; os limites de aceitação são muito bem definidos e o processo de avaliação utilizado hoje é tido, quase que universalmente, como ideal. É possível avaliar esses componentes não só quanto ao seu processo de fabricação, mas também como produto final.

Quando falamos em qualidade de software, muitos autores acreditam que há espaço para melhorias. Os padrões para o desenvolvimento de software seguro normalmente recomendam ou exigem certas técnicas a fim de reduzir os riscos que poderiam vir a causar acidentes ou danos. Essas recomendações normalmente são divididas em níveis de rigor, onde

as mais exigentes são aplicadas nos casos onde as consequências de falhas são mais severas [5]. Apesar de alguns autores acreditarem que somente um processo certificado não garante de fato um produto seguro, essa abordagem apesar de não ser ideal é amplamente aplicada e usada comercialmente, pois, na prática, é o que tem se mostrado factível. Existem interesse e espaço na melhoria de todo o processo de garantir software seguro, para que no futuro se tenha regras tão sólidas e de aceitação comum como em outras áreas do conhecimento, como é o caso da engenharia civil [4]. No escopo desse trabalho, iremos nos ater ao que se tem de mais sofisticado comercialmente hoje em dia, mas o autor acredita que essa discussão seja bastante válida.

Um ponto de extrema importância nos processos de certificação de *software* é a atenção dada à documentação durante todo desenvolvimento dos sistemas relativos à segurança. Ela permite aos avaliadores identificarem as precauções adotadas pelos desenvolvedores para atender os requisitos de segurança. Para atender normas como a IEC 61508, que possui um grande nível de complexidade e uma quantidade imensa de requisitos, é bastante comum o uso de ferramentas digitais que auxiliem o gerenciamento desses artefatos [02].

3. NORMA IEC61508

No decorrer dos próximos subcapítulos entenderemos melhor a norma na qual esse trabalho tem como base. Será apresentada a sua estrutura, e veremos em detalhes as modificações e atualizações em relação a sua edição anterior. Em seguida é feita uma avaliação crítica da IEC61508, e ao final é exposta a estratégia para se alcançar segurança funcional, baseado nesse padrão. A terceira parte da norma, a respeito dos requisitos de software ganha neste trabalho um capítulo a parte.

3.1.IEC

A *International Electrotechnical Commission* (IEC) é, segundo seu próprio site [1], uma instituição líder mundial na preparação e divulgação de padrões internacionais de tecnologias E/E/EP. É baseada em cooperação internacional, e é parte da tríplice de organizações (IEC, ISO, ITU) responsável pela criação de padrões internacionais.

3.1.1. IEC61508

A partir de 1985, a IEC começou uma iniciativa para o desenvolvimento de um padrão genérico para ser utilizado em sistemas E/E/EP relacionados à segurança, diferenciando das abordagens de setores específicos na época (como aviação ou uso em usinas nucleares). Essa iniciativa deu origem, somente em 2000, a primeira edição da norma IEC 61508, dividida em 7 partes. Em 2005 foi publicado um adendo, IEC61508-0, meramente informativo, servindo de relatório técnico. A partir de 2002 se iniciou um processo de revisão, que teve sua edição 2.0 publicada em abril de 2010, na qual o presente trabalho se baseia. [3], [6].

IEC61508 tem se mostrado de grande sucesso e se destaca por abordar todos os componentes dos sistemas relacionados à segurança, bem como todas as etapas do processo, do planejamento e desenvolvimento até o desligamento e desativação desses sistemas. [6]

A terceira parte da norma é relativa aos requisitos de software. Bastante extensa, incluindo seus anexos possui mais de 100 páginas. Nela são apresentados ciclos de vida, tanto de segurança funcional do sistema como um todo, e em detalhe o ciclo de vida de segurança de software. De acordo com cada etapa, são apresentadas as técnicas e medidas que devem ou não ser utilizadas, de acordo com cada nível de SIL. Essas técnicas são classificadas da seguinte maneira:

- HR: Altamente Recomendada. Caso não seja utilizada, deve ser propriamente justificada e aceita pelo órgão avaliador.
- R: Recomendada. Não é obrigatória, porém é sugerida.
- NR: Não Recomendada. Sua utilização implica também na necessidade de uma justificativa a ser aceita.
- --: Indiferente. Não há recomendação nem a favor, nem contrária a sua utilização.

3.2. Objetivos

A norma IEC 61508 foi criada para atender as necessidades relacionadas à segurança em sistemas nos quais as falhas podem causar danos não só a seres vivos, bem como equipamentos ou até o meio ambiente. Seu enfoque é ser um padrão genérico, porém possui complementos para a cobertura de inúmeras áreas específicas.

Sua metodologia visa [2]:

- Fornecer uma abordagem baseada em análise de riscos para definir e estruturar as exigências de desempenho de Sistemas Relacionados à Segurança.
- Incentivar o crescimento de tecnologias relacionados a segurança, apresentando um padrão genérico, e permitindo sua especialização para setores específicos.
- Através de técnicas bem fundamentadas, fazer uso de uma abordagem sistemática e que seja flexível para adaptações futuras.
- Obter um aumento de confiabilidade, tanto de operadores como usuários em geral, em sistemas computadorizados com certificação em segurança.
- Facilitar a criação de métodos para a avaliação e verificação de conformidade a requisitos de segurança.
- Baseando-se em conceitos genéricos, definir requisitos que permitam a criação de uma rede eficiente de fornecimento de componentes a Sistemas Relacionados à Segurança em diversas áreas.
- Permite o uso de diferentes graus de exigência, para a cobertura de 4 Níveis de Integridade de Segurança (SIL).
- Abordagem diferenciada para os modos de operação dos sistemas em Low e High Demand.

3.3. Estrutura

A norma IEC 61508 é dividida em 7 partes:

- 0 Functional safety and IEC 61508
- 1 General requirements
- 2 Requirements for electrical/electronic/programmable electronic safety-related systems
- 3 Software requirements
- 4 Definitions and abbreviations
- 5 Examples of methods for the determination of safety integrity levels
- 6 Guidelines on the application of parts 2 and 3
- 7 Overview of techniques and measures

As partes 1, 2 e 3 contêm todas as exigências da norma. Juntamente com a parte 4, compõe o set básico das publicações de segurança. As partes 0, 5, 6, e 7 são complementares, e não contêm exigências normativas.

Existem publicações para setores específicos, que são especializações da norma, e estão em conformidade com a IEC 61508:

- IEC 62061: Utilizada para Maquinário
- IEC 61511: Especializada em processos
- IEC 61513: Direcionada a uso em Energia Nuclear

Esses padrões são baseados nos princípios de criação de subsistemas contidos na IEC 61508, e levam em consideração as práticas desses setores, bem como sua terminologia e medidas específicas [3].

3.4. Edição revisada 2.0 de 2010

O processo de revisão da norma IEC61508 começou em 2002 estendendo-se até 2010. Nesse processo são seguidas as seguintes etapas [3]:

1. Requisição da visão dos Comitês Nacionais quanto à norma.
2. Baseado nas opiniões do item 1 é preparado um esboço (*Comitee Draft*) para ser avaliado pelos Comitês Nacionais.
3. A partir da avaliação dos comitês é preparado um esboço para votação (*Committee Draft for Vote*)
4. Se for aceito, é preparada uma versão final das mudanças (*Final Committee Draft International Standard*). Elas somente serão publicadas se, novamente, for aprovada em votação.

Um fator com peso considerável na revisão é se, de fato as alterações agregam valor a norma, tendo em consideração também o impacto econômico dessas mudanças. No decorrer deste capítulo veremos algumas dessas mudanças aplicadas nesta nova edição.

- Terminologia: houve mudanças em algumas definições utilizadas na norma, com impacto direto na leitura e interpretação desses termos. Como exemplos nós temos a adição de *subsystem*, que possuía utilização inconsistente; *dangerous failure*; *safe failure*, *element* e *element safety function*.
- Modos de Operação: foram modificados os critérios que classificavam a operação de *high* e *low demand* de Funções de Segurança: foram retiradas as exigências de *proof test frequency*.
- Security: agora as ações intencionais malélicas e não autorizadas devem ser consideradas na fase de análise de Riscos e Hazards. Se esses ataques forem previsíveis, medidas de segurança deverão ser tomadas. Essa medida mostrou-se importante para entrar em conformidade com outros padrões, tanto da IEC como da ISO.
- Especificação de Exigências de E/E/PE: O processo de foi dividido em duas etapas, necessitando agora duas especificações:
Desenvolvimento da especificação de E/E/PE system safety requirements na parte 1 da norma.
Desenvolvimento da especificação de E/E/PE system design requirements na parte 2 da norma.
- Comunicações Digitais: Nas arquiteturas de comunicação tipo White Channel, todo o canal de comunicação (inclusive protocolo, serviços e rede) deve estar em conformidade com as normas IEC61508, IEC 61784-3 ou IEC 62280. Nas do tipo Black Channel, a conformidade das interfaces deve ser com as normas IEC 61784-3 ou IEC 62280.
- Gerenciamento de Segurança Funcional: houve uma reestruturação nesse quesito, e agora é mandatória a nomeação de responsáveis para cada fase, bem como a identificação de todos os envolvidos em determinadas atividades relacionadas diretamente à busca de segurança no desenvolvimento dos sistemas.

- Circuitos Integrados e ASICS: novas exigências foram incluídas, sendo adicionadas novas técnicas. Dois anexos à norma foram complementados, contemplando as mudanças necessárias.
- Manual de segurança para Itens em Conformidade: Nessa revisão foi adicionado o conjunto de exigências para os fornecedores que alegam que seus produtos estão em conformidade com a norma. Esse manual deve conter toda a informação para que esse item em conformidade possa ser integrado com um Sistema Relacionado à Segurança.
- Software IEC 61508-3: podemos listar as seguintes mudanças na parte 3 da norma, relativa à software:
Como saídas de cada fase do ciclo de vida, foram adicionadas propriedades desejadas, como: completude, corretude e previsibilidade.
Mecanismos para a reutilização de elementos de software que originalmente não foram desenvolvidos para segurança, através da demonstração de segurança em outras aplicações.
As técnicas utilizadas no desenvolvimento de software presente nos Anexos A e B da norma foram revistos, removendo as entradas obsoletas ou de uso incomum, sendo substituídas por métodos mais comuns atualmente.

3.5. Considerações sobre a norma

A norma IEC 61508 é tida como o melhor padrão genérico para desenvolvimento de Sistemas Relacionados à Segurança [7], se destacando por abordar todos os componentes destes, bem como todas as etapas do processo: do planejamento e desenvolvimento até o desligamento e desativação desses sistemas [6]. Sua abordagem é extremamente exigente nos seus requisitos. Possui uma análise de riscos muito aprofundada, o que na prática acaba levando a uma redução de custo total do processo de desenvolvimento e certificação para as companhias. Esse nível de exigência também beneficia a relação entre desenvolvedores e clientes: devido a especificações mais precisas, o entendimento entre as duas partes é facilitado[6].

Quando comparado a outras padrões que abordam o mesmo assunto, IEC 61508 tem vantagem ao ser ser mais prescritiva e focada na qualidade do produto; porém é ainda atrasado se considerarmos códigos de outros campos da engenharia, como a engenharia civil [4].

Uma liberdade oferecida pela norma é a utilização de técnicas não recomendadas (*NR*), ou ainda a não utilização de técnicas apontadas com altamente recomendadas (*HR*) para o desenvolvimento de software. Nesses casos, é necessária uma justificativa para tais atitudes: porém há dificuldade em determinar quais justificativas são aceitas pela entidade avaliadora, em cada caso. [2]

Como desvantagens, destaque-se o volume monstruoso da norma e a dificuldade na sua leitura e interpretação [6], por ser bastante textual, nem sempre expressa de maneira precisa e estruturada. Também apresenta deficiências para aplicação em alguns contextos específicos [7]. A complexidade e o alto custo do processo de certificação acaba afastando empresas de menor porte a buscar certificações de segurança [6].

Outro ponto muitas vezes criticado diz respeito quanto ao emprego dos níveis de integridade de segurança. A complexidade que envolve os sistemas muitas vezes dificulta ou praticamente impossibilita uma estimativa precisa de SIL. Alia-se a isso o fato de que a interpretação do SIL apresenta divergências entre normas, além dos cálculos estimativos serem feitos baseados em considerações sobre confiabilidade [2].

Alguns autores [5] acreditam que, apesar da norma possuir uma série de orientações e servir de guia, ela acaba sendo mais focada na qualidade e repetibilidade do processo, do que, de fato em garantir propriedades de segurança.

3.6. Alcançando Segurança Funcional

A estratégia para o alcance de segurança funcional tem como base [3]:

- Gerenciamento de Segurança Funcional
- Exigências técnicas para as fases dos ciclos de vida relacionados à segurança
- Avaliação de Segurança Funcional
- Nível de competência de indivíduos das equipes de desenvolvimento.

A norma utiliza uma abordagem baseada na análise de riscos, e para se alcançar um determinado Nível de Integridade de segurança, cada Função de Segurança deve ser tal que:

- A frequência de falhas dos Sistemas Relacionados à Segurança é suficientemente baixa para que a frequência de eventos que possam causar hazard esteja de acordo com o nível de risco e/ou
- O Sistema Relacionado à Segurança possa modificar as consequências do evento que possa causar hazard até um nível de risco tolerável.

É levado em consideração o modo de operação no qual o sistema esteja funcionando. Ele é classificado da seguinte maneira [3][7]:

- Baixa demanda: quando a demanda por operação da Função de Segurança não ultrapassa 1 por ano, ou maior que a frequência de testes. Tem sua medida na possibilidade de falha quando ativada.
- Alta demanda ou modo contínuo: quando a ativação é mais comum, tendo sua medida baseada na taxa de hazards por hora de funcionamento.

3.7. Ciclos de Vida

Estudos mostram [3] que a abordagem na criação de Sistemas Relacionados à Segurança deve cobrir todas as fases de desenvolvimento. Isso se deve ao fato de diversas etapas mostram ter parcelas significativas na origem das falhas nos sistemas de controle. A figura 3.1 demonstra essa distribuição:

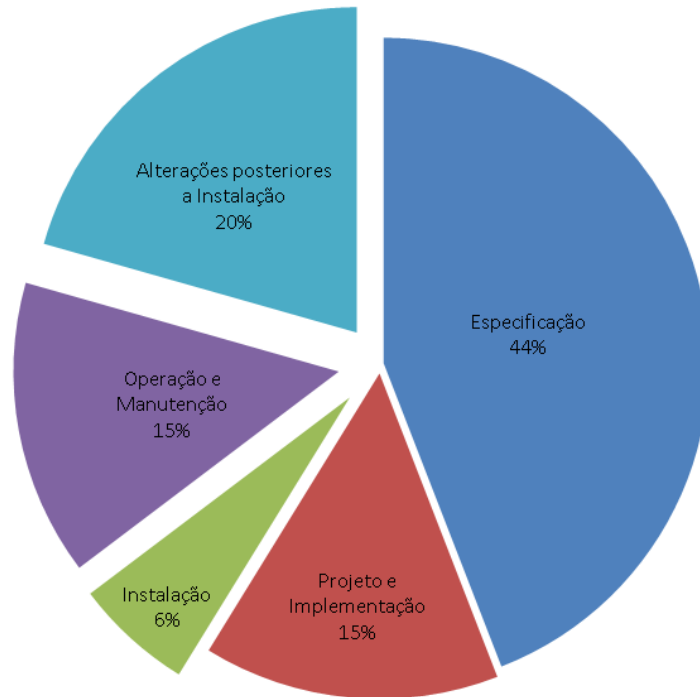


Figura 3.1: Origem das causas primárias de falhas em sistemas de controle [3]

É possível perceber que a maioria das falhas são incorporadas aos sistemas durante o seu desenvolvimento, durante as fases de Especificação e Projeto e Implementação.

Com a finalidade de fornecer uma cobertura total, a norma IEC61508 apresenta 3 ciclos de vida importantes para lidar com Sistemas relacionados à Segurança. Elas serão detalhadas a seguir:

- Ciclo de Vida de Segurança Funcional no capítulo [3.8](#)

Ciclo de Vida de Segurança de Sistemas E/E/PE não é detalhado neste trabalho, e se refere à IEC 61508-2.

- Ciclo de projeto e desenvolvimento de Software no capítulo [4.3](#)

3.8. Ciclo de Vida de Segurança Funcional

Para lidar de maneira sistemática com as atividades necessárias para se alcançar um nível de segurança almejado, a norma adota um Ciclo de Vida de Segurança Funcional, visto na Figura 3.2. Este é iniciado com definição do conceito, definição geral do escopo. Depois é feita a análise de *hazards* e riscos. Juntas, essas atividades determinam *o que deve ser feito* (derivação dos requisitos de segurança) e o nível de segurança que deve ser alcançado (SIL) [7]. As 16 fases serão vistas com um nível maior de detalhe no decorrer deste capítulo [2].

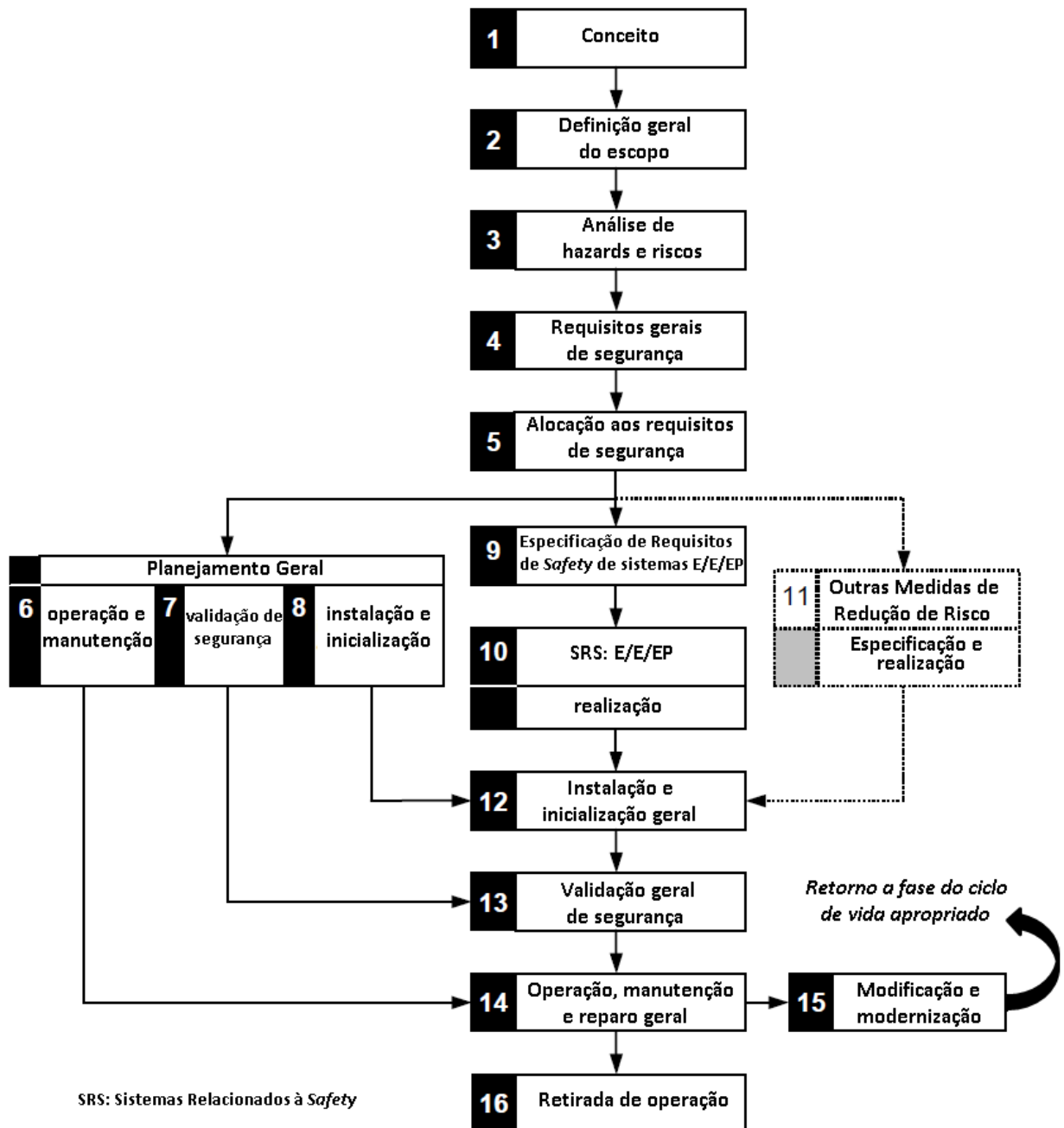


Figura 3.2: Ciclo de vida de Segurança Funcional.

- **Conceito:** a fase inicial visa obter um entendimento geral do sistema e do ambiente relacionado. Deve ser satisfatório a ponto de possibilitar a execução correta das próximas fases do ciclo de vida.
- **Definição geral do escopo:** a fase de definição de escopo tem por finalidade estabelecer a limitação do sistema de controle, bem como do sistema que estará sendo controlado. Auxilia na especificação do escopo das análises de *riscos* e de *hazards*.
- **Análise de hazards e riscos:** a etapa de análise de *hazards* e riscos visa definir quais os riscos e eventos de *hazard* que possam ocorrer em determinado sistema. Isto se aplica em todas as situações passíveis de previsão e para os diversos modos de operação.

- **Requisitos gerais de segurança:** nesta parte do ciclo de vida devem ser especificados e desenvolvidos os requisitos gerais de segurança. Inclui os requisitos ligados à integridade de segurança e funções de segurança.
- **Alocação aos requisitos de segurança:** fase em que são feitas as alocações necessárias das funções de segurança, definidas nos requisitos gerais de segurança, para atingir o nível desejado de segurança funcional. As funções são alocadas aos sistemas de controle, sistemas externos de redução de risco e sistemas com outras tecnologias que estejam relacionadas à segurança.
- **Planejamento de operação e manutenção:** fase voltada ao planejamento dos processos de operação e manutenção de um sistema, visando assegurar a conservação das condições gerais de segurança ao longo destes processos.
- **Planejamento de validação de segurança:** esta etapa tem como finalidade a elaboração de uma estratégia que auxilie na aprovação da segurança do sistema como um todo.
- **Planejamento de instalação e inicialização:** esta fase tem como objetivo formular um plano para que tanto a instalação quanto a inicialização do sistema ocorram de forma controlada, para assegurar que sejam atingidas as metas de segurança funcional.
- **Especificação dos requisitos de segurança de sistemas E/E/EP:** durante essa fase são definidos os requisitos de segurança para os sistemas E/E/EP, de acordo com as exigências para se alcançar o Nível de Integridade de Segurança desejado.
- **Sistemas relacionados à Segurança – Realização:** a fase de realização tem por finalidade a elaboração do sistema responsável por efetuar o controle, de acordo com os requisitos levantados durante a fase apresentada no item anterior, tanto com relação à integridade de segurança, quanto para as funções de segurança.
- **Outras Medidas de Redução de Risco:** fase que tem como meta a inclusão de outras medidas e sistemas externos que atuem na redução de riscos (não pertence ao escopo da IEC 61508).
- **Instalação e inicialização geral:** fase onde é realizada a instalação e inicialização do sistema, seguindo os processos definidos anteriormente.
- **Validação geral de segurança:** nesta fase é executada a verificação do sistema de controle, para assegurar que este satisfaça as especificações globais de segurança (nível de integridade de segurança e funções de segurança). Considera a alocação, anteriormente estabelecida, para os requisitos de segurança.
- **Operação, manutenção e reparo geral:** fase que tem por objetivo assegurar que a segurança funcional do sistema se mantenha durante os processos de operação, manutenção e reparos eventuais do sistema.
- **Modificação e modernização:** Esta etapa tem como responsabilidade assegurar a manutenção da segurança funcional, seja antes ou depois de modificações feitas no sistema.
- **Retirada de operação:** Etapa que tem como foco assegurar que as atribuições de segurança funcional sejam satisfatórias ao longo do processo de retirada de operação do sistema, bem como ao término deste processo.

4. IEC61508-3

Neste capítulo veremos em detalhe a parte da norma que cobre os requisitos de software.

4.1. Segurança de Software

Software para Sistemas Relacionados à Segurança deve ser desenvolvido de tal maneira que satisfaça os requisitos de segurança definidos para ele. Para permitir posteriormente a validação e verificação da entidade certificadora, é crucial que se mantenha o acompanhamento entre os requisitos de segurança e as decisões tomadas no seu desenvolvimento [7].

4.2. Ciclo de vida de Segurança de Software

O processo de desenvolvimento de segurança de software é organizado em etapas como na Figura 4.1. Dependendo do SIL almejado, a norma IEC61508 faz diferentes exigências, em cada uma dessas etapas. As fases podem conter subfases, e são detalhadas no decorrer deste capítulo. [2]

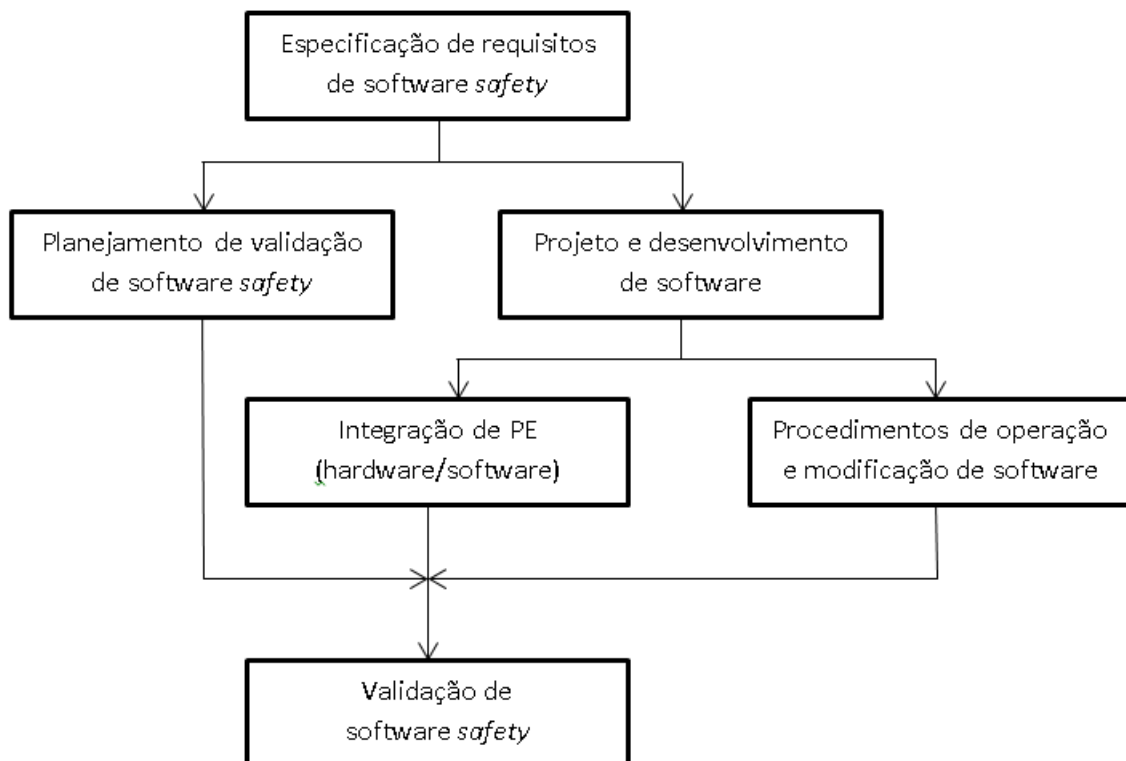


Figura 4.1: Ciclo de vida de Segurança de Software

- **Especificação dos requisitos Segurança de Software:** A especificação de requisitos de Segurança de Software deve abranger a definição de quais as Funções de Segurança serão executadas, assim como o modo de operação e quais os níveis de integridade de cada uma das Funções de Segurança.

- **Plano de validação de Segurança de Software:** o plano de validação de Segurança de Software deve apresentar a estruturação do gerenciamento de segurança, as atividades relativas à segurança e também os marcos de aprovação presentes no ciclo de vida. Deve ser elaborado no início do ciclo de vida, sofrendo revisões caso sejam feitas mudanças no sistema inicial.
- **Projeto e desenvolvimento de software:** esta fase é dividida em subfases e será analisada em detalhe no capítulo [4.3](#).
- **Integração (hardware/software):** a etapa de integração visa comprovar que a interação entre hardware e software durante o desempenho de determinada função ocorre de forma correta. No caso de softwares com SIL superior a zero, é necessária a elaboração de um plano de teste de integração entre software e hardware no começo do ciclo de desenvolvimento.
- **Procedimentos de operação e manutenção de software:** esta fase tem como objetivo definir procedimentos para assegurar a conservação de segurança funcional ao longo dos processos de operação e manutenção do software. Devem-se estabelecer condutas a serem seguidas quando da ocorrência de falhas de software, bem como procedimentos para diagnosticar os defeitos. Igualmente, é necessário a definir processos para revalidação e quais os requisitos do relatório de manutenção.
- **Validação de segurança de software:** consiste em verificar se cada fase do ciclo satisfaz seus requisitos específicos de segurança, identificado nas fases anteriores. Ou seja, é a validação do software contra sua especificação de requisitos de segurança. A análise de segurança e os testes adequados devem ser executados e documentados, com as evidências de que o software cumpre os requisitos apresentadas em um documento de justificativa de segurança.

4.3.Ciclo de projeto e desenvolvimento de Software

A fase de “Projeto e desenvolvimento de software” mencionada no capítulo anterior, pode ter as suas subfases visualizadas na Figura 4.2, representando o modelo de desenvolvimento em V. São aceitos outros modelos pela entidade, porém este é o mais usual.

Cada fase desse modelo é detalhada nos itens a seguir[2]:

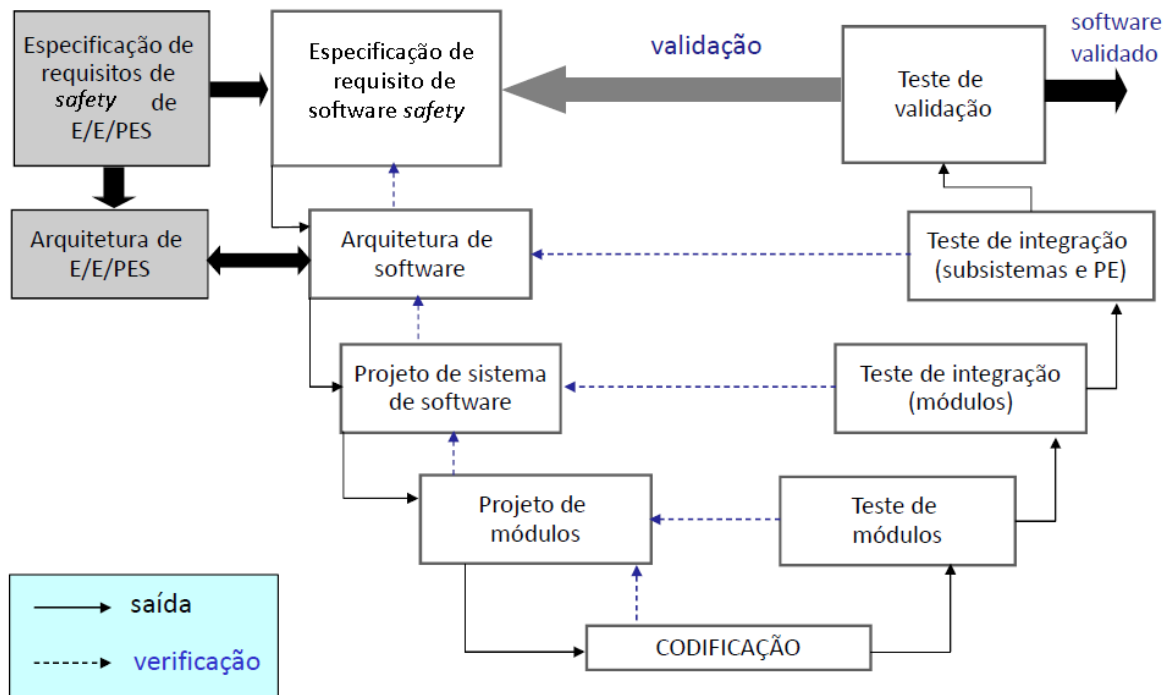


Figura 4.2: Modelo V de desenvolvimento de software. Adaptado de (WEBER)

- **Arquitetura:** A fase de arquitetura tem por objetivo elaborar a arquitetura de software visando cumprir os requisitos de segurança definidos, de acordo com o nível de integridade de segurança solicitado. Também leva em consideração requisitos que tenham sido inseridos no software através da arquitetura do hardware e as interações software/hardware que possam ocorrer.
- **Suporte a ferramentas e linguagens de programação:** O objetivo desta etapa é apurar um grupo de ferramentas que irão assessorar os processos de avaliação, validação, verificação e modificação do software. Estas ferramentas, dentre as quais estão os compiladores e linguagens de programação, devem ser apropriadas para o nível de integridade de segurança desejado.
- **Projeto de sistema:** a meta desta etapa é elaborar o projeto de um software que seja passível de verificação, análise e que mantenha os atributos de segurança durante processos de modificação. Além disso, deve satisfazer os requisitos de segurança de software especificados para o nível de *Safety Integrity Level* exigido.
- **Projeto de módulos:** esta fase tem como finalidade a análise e divisão do software em módulos, refinando o projeto de arquitetura inicial de acordo com os subsistemas identificados. Da mesma forma, deve ser especificado o projeto de cada módulo individualmente, bem como os testes a serem efetuados em um determinado módulo do software.
- **Codificação:** ao longo da etapa de codificação, devem ser utilizadas normas que contenham boas práticas de codificação, especificando regras que restringem o uso de estruturas da linguagem, quando empregadas em módulos com requisitos de segurança. Ainda, devem ser usadas ferramentas que detenham um certificado de validação com reconhecimento internacional. A linguagem de programação deve ser compatível com os métodos de desenvolvimento e possuir mecanismos que auxiliem na identificação de incorreções no programa.

- **Teste de módulos:** esta etapa é responsável pelo teste de cada módulo do software, visando verificar se cada um destes executa as funções para as quais foram projetados e que satisfazem as definições exigidas de integridade e funções de segurança.
- **Teste de integração:** esta fase tem como finalidade verificar o cumprimento dos requisitos de segurança estabelecidos para o software. A validação é feita através de procedimentos que comprovem a interação correta e adequada entre os módulos, subsistemas e seus componentes. As atividades de teste de integração devem ser elaboradas concorrentemente ao longo das etapas de projeto e desenvolvimento.

4.4.Artefatos

Cada fase do ciclo de desenvolvimento gera um ou mais subprodutos. Estes artefatos gerados ao final de cada fase dos ciclos são utilizados como “matéria prima” para a fase posterior, ou seja, o *output* de uma fase é o *input* para a execução da próxima fase, formando uma cadeia. No contexto da IEC 61508, estes podem ser de variados formatos como, por exemplo, especificações, descrições, diagramas, série de instruções, listas, *log* de atividades, plano, relatório [7]. O gerenciamento dessa série de itens é um desafio durante os projetos.

4.5.Backtracking

Durante a execução de projetos, é comum que algumas necessidades e detalhes do sistema desenvolvido só sejam percebidos em fases mais avançadas. Para manter-se a conformidade e não haver impacto nos requisitos de segurança, essas modificações do projeto inicial devem ser analisadas, e fases do ciclo de vida que já eram tidas como finalizadas devem ser revistas. Essa dinâmica é chamada de *backtracking*.

5. IMPLEMENTAÇÃO

O desenvolvimento de software voltado a certificações é um processo extenso e bastante complexo. Este capítulo aborda a ferramenta digital desenvolvida que tem por fim auxiliar equipes de desenvolvimento e gerenciamento de projetos relacionados à segurança: *Software Safety Certification Sidekick! (SSCSidekKick!)*.

Seu uso é voltado apenas à parte de requisitos de software. No caso da norma IEC61508, cobre apenas o terceiro capítulo, *Software Requirements*. Pode ser utilizada também para outras normas com um fluxo de desenvolvimento similar, necessitando apenas de um arquivo de configuração, e adaptação em um método que analisa a conformidade com a norma.

É digno de nota que a utilização dessa ferramenta ou similares visa auxiliar no processo de desenvolvimento, porém não garante a certificação, pois não há meio determinísticos para tanto. Existe um processo de avaliação pelas entidades, onde as justificativas para a utilização das técnicas podem ou não ser aceitas.

Esta se diferencia de outras caras ferramentas disponíveis no mercado por ser gratuita, sem fins comerciais.

No decorrer deste capítulo serão apresentadas as funcionalidades que o programa oferece, e em seguida entraremos em detalhes como foi feita a sua implementação. Abordaremos sua arquitetura, e as ferramentas utilizadas para o desenvolvimento,

5.1. Funcionalidades

De uma maneira geral, a ferramenta oferece uma plataforma para acompanhar o desenvolvimento de software. Durante cada fase de desenvolvimento, é possível fazer o controle das técnicas e medidas necessárias para se alcançar determinado SIL. Também podem ser adicionados comentários e justificativas para o uso de determinadas técnicas, bem como apontamentos para documentos e artefatos.

A qualquer momento, é permitido ao usuário verificar a conformidade para o nível de integridade desejado, considerando apenas as fases do processo de desenvolvimento que já foram contempladas.

Projetos podem ser exportados e importados para a ferramenta. Quaisquer alterações no projeto são armazenadas para controle administrativo.

A ferramenta suporta inicialmente a norma IEC 61508-3 v2.0 de 2010, porém é facilmente adaptada para atualizações na norma (como mudança nas técnicas, modificação de recomendações ou fases do ciclo de desenvolvimento) ou outras normas com características semelhantes.

5.2. Desenvolvimento da Ferramenta

A ferramenta proposta nesse trabalho foi desenvolvida em Java, uma linguagem de programação orientada a objetos.

5.2.1. Arquitetura

Como arquivos de entrada e saída, tanto para configuração da norma como para o arquivamento de um projeto, foi escolhido o formato XML. Por ser largamente utilizado, de fácil leitura, hierarquizável, e expansível, se mostrou bastante eficiente para os propósitos deste trabalho.

Classes utilizadas:

- Project
 - Atributos básicos: ID, nome,
 - Listas: fases do ciclo de desenvolvimento, regras da norma, exigências para cada nível de integridade, usuários, logs.
- Phase
 - Identificação da fase, apontamento para entregáveis.
- Rule
 - Identificação, campo para justificativa, checkbox se foi ou não utilizada.
- RuleRecom
 - Cada objeto dessa classe representa a recomendação para 1 nível de integridade de segurança para 1 regra. Está diretamente ligado a um objeto da classe Rule.
- User
 - Identificação de um usuário relacionado ao projeto.
- Log
 - Entrada de log.

Principais métodos:

- XML2Obj
 - Converte um arquivo de entrada no formato XML para um objeto, manipulável pela aplicação. Utilizado ao se criar um projeto novo ao importar a configuração de norma, ou ao carregar um arquivo de projeto previamente salvo.
- exportProject
 - O caminho inverso do método anterior: transforma um objeto Project em um arquivo XML.
- alterPhase
 - Permite a mudança do atributo currentPhase do projeto, alterando a atual fase no ciclo de desenvolvimento.
- addlogentry
 - Utilizada cada vez que há uma mudança no projeto, criando uma entrada de Log.
- verificarSIL
 - Recebe o objeto Project, nível de SIL desejado e o nome do standard. No caso da IEC61508-3, varre todos os objetos Rule, fazendo o “casamento” com os objetos RuleRecom que possuam o mesmo ID. São comparadas as recomendações (HR, R, NR, --) referentes a essa técnica, para o nível desejado. No caso de uma técnica NR que tem o seu valor ruleUsed como verdadeiro, ou uma técnica HR com ruleUsed falso, é gerada uma saída avisando que essa técnica deverá ter sua justificativa de uso aceita pelo órgão certificador.
 - Esse método deve ser modificado no uso de outras normas, pois sua logica é somente compatível com a IEC61508-3.

5.2.2. Ferramentas utilizadas

Como plataforma de desenvolvimento, foi utilizado NetBeans IDE 7.2.1, para Java 1.7.0

A criação e manipulação dos arquivos XML foi feita utilizando o editor de texto Notepad++ v6.2.2.

5.2.3. Bibliotecas utilizadas

Para o desenvolvimento da aplicação, foram utilizadas as seguintes bibliotecas:

- Bibliotecas Padrão: *java.util* (para tratamento de Arrays), *java.io* (para tratamento de arquivos e excessões). Para a interface gráfica, podemos citar *javax.swing* e *java.awt*.
- XStream: [11] permite a conversão de objetos em Java em arquivos XML, e vice-versa. Utilizada na importação de norma, e carregamento e salvamento dos arquivos.

5.2.4. Detalhes do Desenvolvimento

Ao se abrir a ferramenta, o usuário tem a sua disposição a seguinte tela, apresentada na figura 5.1:

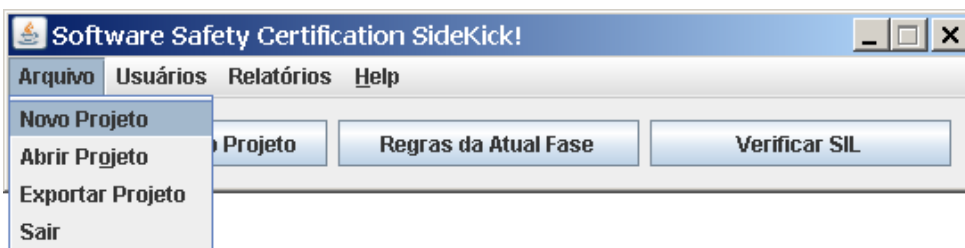


Figura 5.1: Tela inicial do *SSCSideKick!*, com o menu de arquivo selecionado.

Ao se criar um novo projeto, através do menu Arquivo/Novo Projeto, preenche-se as informações básicas e é importado um arquivo de norma, em formato XML. Esse processo gera um objeto da classe *Project*, que é manipulado pelos demais métodos da aplicação. A Figura 5.2 mostra a interface para a criação de um projeto. Vale salientar que o campo “Norma para Certificação” corresponde ao caminho para um arquivo de configuração de norma.

Figura 5.2: Interface de criação de projeto.

Na Figura 5.3 vemos a estrutura de um arquivo de configuração de norma:

```

1  <?xml version="1.0" encoding="UTF-8" ?>
2  <project>
3    <standard></standard>
4    <phases>
5      <phase>
6        <phaseID></phaseID>
7        <phaseDesc></phaseDesc>
8      </phase>
9    </phases>
10   <rules>
11     <rule>
12       <ruleID></ruleID>
13       <ruleDesc></ruleDesc>
14       <ruleRef></ruleRef>
15       <rulePhase></rulePhase>
16       <subRules></subRules>
17     </rule>
18   </rules>
19   <recoms>
20     <rulerecom>
21       <reconruleID></reconruleID>
22       <sevlvl></sevlvl>
23       <recom></recom>
24     </rulerecom>
25   </recoms>
26 </project>
27

```

Figura 5.3: Arquivo de configuração de norma no formato XML.

Funcionalidades oferecidas ao usuário são a visualização e edição dos atributos do projeto. Percebe-se na Figura 5.4 que o campo “Norma para Certificação” não pode ser editado, pois é configurado a partir do carregamento de uma norma, tão pouco o campo abaixo de “Fase Atual”, que identifica o nome da fase no qual o projeto se encontra no momento. Ao se utilizar o botão *Alterar*, o correspondente atributo é modificado no objeto *Project*.

Figura 5.4: Janela para edição de um projeto.

Para o salvamento de um Projeto, ocorre a conversão do objeto *Project* em uma String no formato XML, através do método *exportProject*. Este pode ser importado novamente ao programa, quando se deseja *abrir* um projeto previamente criado.

Podemos observar na figura 5.5 que na estrutura de um arquivo de projeto, a norma fica “embutida”, porém com a adição de alguns campos:

```

1  <<xml version="1.0" encoding="UTF-8" >>
2  <project>
3      <projectID></projectID>
4      <projectName></projectName>
5      <startdate></startdate>
6      <enddate></enddate>
7      <standard></standard>
8      <severitywanted></severitywanted>
9      <currentPhase></currentPhase>
10     <phases>
11         <phase>
12             <phaseID></phaseID>
13             <phaseDesc></phaseDesc>
14             <phaseEndDeliv></phaseEndDeliv>
15         </phase>
16     </phases>
17     <rules>
18         <rule>
19             <ruleID></ruleID>
20             <ruleDesc></ruleDesc>
21             <rulePhase></rulePhase>
22             <ruleRef></ruleRef>
23             <subRules></subRules>
24             <ruleUsed></ruleUsed>
25             <ruleJust></ruleJust>
26         </rule>
27     </rules>
28     <recoms>
29         <rulerecom>
30             <reconruleID></reconruleID>
31             <sevlvl></sevlvl>
32             <recom></recom>
33         </rulerecom>
34     </recoms>
35     <users>
36         <user>
37             <userID></userID>
38             <userName></userName>
39             <userContact></userContact>
40             <userRole></userRole>
41         </user>
42     </users>
43     <logs>
44         <log>
45             <logUser></logUser>
46             <logDesc></logDesc>
47         </log>
48     </logs>
49 </project>
50

```

Figura 5.5: Modelo do arquivo de saída de um projeto no formato XML.

Com um projeto aberto na ferramenta é possível:

- Adicionar um indivíduo ao projeto:
Após o preenchimento dos dados pessoais, é criado um objeto *User* com estas informações e este é adicionado à lista de usuários. A Figura 5.6 representa a interface para a criação.

Figura 5.6: Interface para criação de usuário.

- **Modificação das regras/técnicas:**
Apenas as regras pertencentes a atual fase do ciclo de desenvolvimento são disponíveis. É possível selecionar se a técnica foi ou não utilizada, e adicionar algum comentário como justificativa do motivo do uso. É possível utilizar esse campo para apontar para algum documento externo, se for adequado. A Figura 5.7 exemplifica a visualização de regras, e a Figura 5.8 exibe a interface para a edição de uma regra, na qual apenas os campos de justificativa e utilização são passíveis de mudança. Utilizando o botão salvar, é alterado o objeto *Rule* referente a regra correspondente.

| ID | Descrição da Regra | Referência | SubRegras | SIL1 | SIL2 | SIL3 | SIL4 | Utilizada | Justificativa |
|----|--|--------------|-----------|------|------|------|------|-------------------------------------|----------------------------------|
| 11 | Semi-formal methods | Table B.7 | | R | R | HR | HR | <input checked="" type="checkbox"/> | |
| 12 | Formal methods | B.2.2, C.2.4 | | -- | R | R | HR | <input checked="" type="checkbox"/> | |
| 2 | Forward traceability between the system safety requirements and the software safety requirements | C.2.11 | | R | R | HR | HR | <input checked="" type="checkbox"/> | |
| 3 | Backward traceability between the safety requirements and the perceived safety needs | C.2.11 | | R | R | HR | HR | <input type="checkbox"/> | Utilizada uma regra equivalente. |
| 4 | Computer-aided specification tools to support appropriate techniques/measures above | B.2.4 | | R | R | HR | HR | <input checked="" type="checkbox"/> | |

Figura 5.7: Visualização de técnicas da atual fase

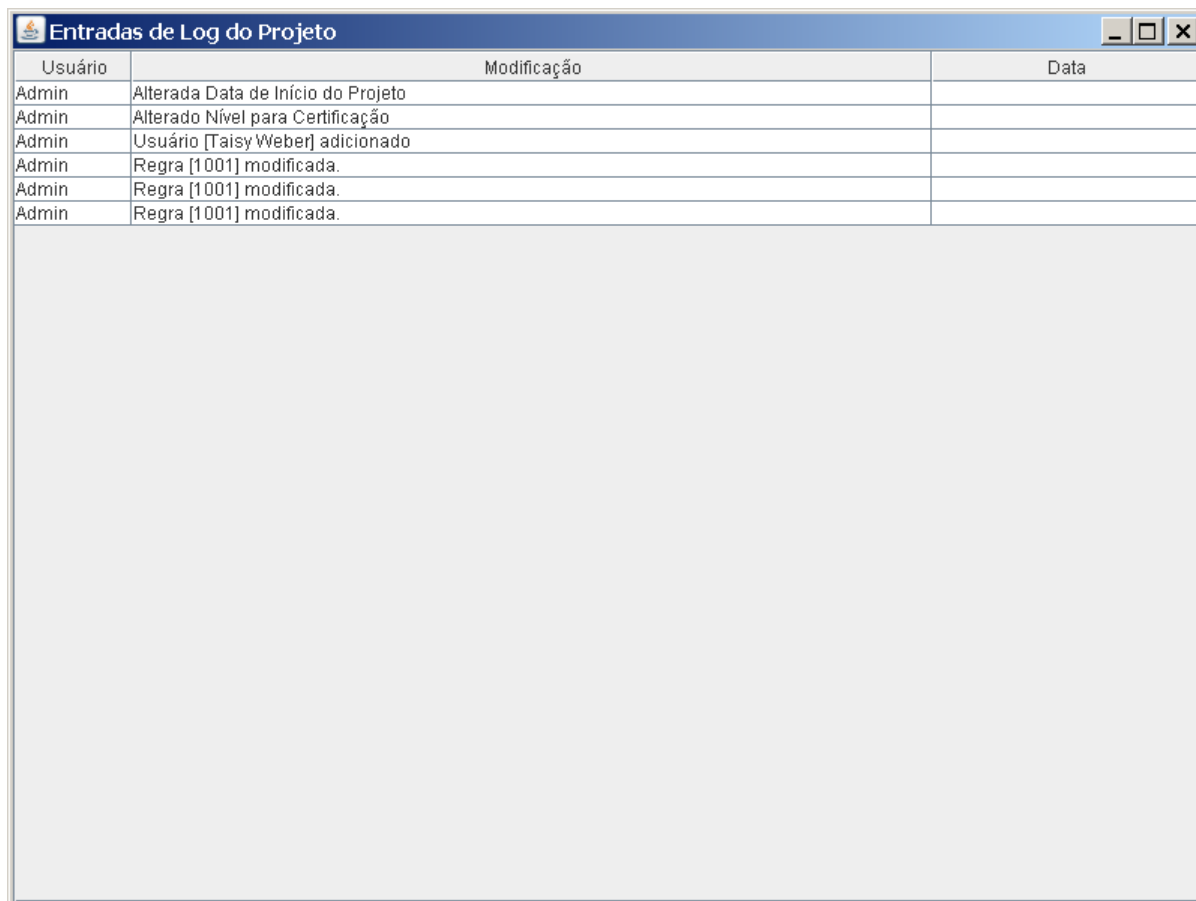
Figura 5.8: Interface para alteração de regra.

- **Mudança de Fase no Ciclo de desenvolvimento.**
O usuário tem a capacidade de modificar a fase na qual o projeto se encontra no momento. Essa mudança causa o seguinte impacto: apenas as regras e técnicas da nova fase ficarão disponíveis para edição, e a função de Verificar SIL também será alterada, para corresponder a atual fase.
- **Verificar SIL**
Essa função fica disponível durante todas as fases do ciclo de desenvolvimento. Ao ser utilizada, a aplicação varre todas as fases até a atual e verifica quais das técnicas devem ter seu uso (ou a falta dele) com uma justificativa aceita pela entidade certificadora, para se alcançar o SIL almejado. São destacadas as técnicas classificadas como NR (*not recommended*) que foram utilizadas, bem como as HR (*highly recommended*) que foram marcadas como não utilizadas. A Figura 5.8 demonstra uma saída típica dessa funcionalidade.

Figura 5.8: Verificação de SIL.

- Ver logs

Ao se fazer qualquer alteração no projeto que está sendo trabalhado, é gerada uma entrada de log com a identificação do que foi mudado, por quem e quando. Estas entradas ficam armazenadas na lista de logs, podendo ser acessadas para conferência, como na Figura 5.9.



| Usuário | Modificação | Data |
|---------|------------------------------------|------|
| Admin | Alterada Data de Início do Projeto | |
| Admin | Alterado Nível para Certificação | |
| Admin | Usuário [Taisy Weber] adicionado | |
| Admin | Regra [1001] modificada. | |
| Admin | Regra [1001] modificada. | |
| Admin | Regra [1001] modificada. | |

Figura 5.9: Arquivo de log.

6. CONCLUSÃO

O desenvolvimento deste trabalho possibilitou um entendimento da notável norma IEC61508, e uma revisão dos conceitos básicos de segurança. Para a implementação da ferramenta, foi praticada a programação no paradigma de Orientação a Objetos. A ferramenta final, apesar de limitada, cumpre seu papel de auxiliar o desenvolvimento de software seguro, sendo adaptável, e gratuita.

Como dificuldades encontradas, podemos citar a limitação da biblioteca XStream em manipular estruturas do tipo *Array* contendo outros *arrays*. Isso não permitiu que os atributos de recomendações para cada nível de severidade fossem encapsulados na estrutura *rule*, para que a ferramenta pudesse receber normas com um diferente número de severidades. Essa limitação foi contornada, se criando uma lista de recomendações que apontassem para a regra a qual ela se referia, porém deixando a saída em XML menos legível.

A inexperiência do autor na utilização da norma também é refletida nos requisitos e funcionalidades desejadas, que é baseado grande parte em pesquisa.

Para trabalhos futuros, são sugeridas algumas novas funcionalidades:

- Sistema de Acessos: Sistema de usuários e senhas, com permissões e acessos diferenciados, permitindo um maior controle de projeto.
- Sistema de notificação para mudanças nas fases de desenvolvimento: Mudanças que impactam o projeto são notificadas aos usuários por e-mail, SMS.
- Perspectivas em Alto Nível e em Detalhe das Fases: Visualização em alto nível dos ciclos de vida do projeto, bem como a explosão para menor nível de granularidade. Desde a visão geral do projeto até a listagem de técnicas necessárias para qualquer subfase de desenvolvimento.
- Backtracking: Modificações que impactem fases passadas de projeto são sinalizadas, indicando quais fases devem ser revistas para que a conformidade à norma seja respeitada.

Outras possibilidades contam com o desenvolvimento de uma ferramenta com as mesmas funcionalidades, porém para outras plataformas:

- Web: permitindo o acesso através de um *browser* a partir de qualquer computador conectado a internet.
- Aplicativo para celulares: usuários poderiam ter acesso à ferramenta a partir do uso de *smartphones*.
- Add-on para plataformas de desenvolvimento: a integração com ferramentas que possuem repositório de código e documentação abre um grande leque de possibilidades. As funcionalidades dessas *suítes* como controle de acesso a documentos, versionamento, repositório de código, geração de relatórios poderiam deixar a aplicação muito completa.

A participação de pessoas mais experientes em relação ao uso da norma em projetos práticos poderia também identificar melhorias e novas funcionalidades.

BIBLIOGRAFIA

- [1] **International Electrotechnical Commission** <<http://www.iec.ch>>. Acessado em 08/06/2012
- [2] **Bandeira, Diego C.: Aplicação da Norma IEC 61508 em Sistemas Críticos.** 2001. 39 f. Projeto de Diplomação (Bacharelado em Ciência da Computação) – Instituto de Informática, UFRGS, Porto Alegre.
- [3] **Bell, Ronn: Introduction and Revision of IEC 61508** ACS Workshop on Tools and Standards, Conference in Research and Practice in Information Technology, Vol. Nº 55, 2005.
- [4] **Wassyng, A., Maibaum, T., Lawford, M., Bherer, H.: Software Certification: Is There a Case against Safety Cases?** Monterey Workshops 2010, LNCS 6662, pp. 206–227, 2011.
- [5] **MCDERMID, J.A. Software Safety: Where’s the Evidence?** Proc. 6th Australian Workshop on Industrial Experience, v.3, 2001.
- [6] **FALLER, R. Project Experience with IEC 61508 and Its Consequences** U. Voges (Ed.): SAFECOMP 2001, LNCS 2187, v. 2187 p. 200–214, 2001.
- [7] **Panesar, Rajwinder K., Sabetzadeh, M., Briand, L.: Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard.** 2010 Third International Conference on Software Testing, Verification and Validation
- [8] **SIQUEIRA T. F.; MENEGOTTO, C.C.; WEBER, T. S.; NETTO, J.C.; WAGNER, F.R. Desenvolvimento de Sistemas Embarcados para Aplicações Críticas IV** Escola Regional de Redes de Computadores, Passo Fundo, setembro 2006.
- [9] **Weber, Taisy S.: Slides de Aula da Disciplina de Fundamentos de Tolerância à Falhas**
- [10] **Extensible Markup Language (XML)** <<http://www.w3.org/XML/>>. Acessado em novembro de 2012.
- [11] **XStream** <<http://xstream.codehaus.org/>>. Acessado em novembro de 2012.
- [12] **Cechin, Sérgio Luis, Weber, Taisy Silva, Netto, J. C. Arquiteturas Moon(D) para portas de entrada e saída de remotas em conformidade com a IEC 61508** In: Congresso Brasileiro de Automática, 2012, Campina Grande, PB.19. Congresso Brasileiro de Automática. Campinas, SP: Sociedade Brasileira de Automática, 2012. v.1. p.4500 – 4507
- [13] **Weber, Taisy Silva, Cechin, Sérgio Luis, Netto, João César Integridade de segurança em sistemas críticos de controle e instrumentação** In: WES - Workshop em Sistemas Embarcados para Indústria de Óleo, Gás e Energia, 2012, Rio Grande, RS. Anais do WES 2012: Workshop em Sistemas Embarcados para Indústria de Óleo, Gás e Energia. Rio

Grande - RS: FUNDAÇÃO UNIVERSIDADE FEDERAL DO RIO GRANDE, 2012. v.1.
p.1 - 4

[14] **Rede E3** <<http://projeto3.ece.ufrgs.br/>> acessado em novembro de 2012.

ANEXO <ARTIGO TG1: DESENVOLVIMENTO DE SOFTWARE EMBARCADO DE ACORDO COM A NORMA IEC-61508 ED 2.0 (2010)>

Desenvolvimento de Software Embarcado de Acordo com a Norma IEC-61508 Ed 2.0 (2010)

Lukas Lampert, Taisy Silva Weber (Orientadora)

Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre – RS – Brasil

{llampert,taisy}@inf.ufrgs.br

Resumo. *A utilização de sistemas eletrônicos é cada vez mais comum nos dias de hoje, estando presentes nas mais diversas áreas. De grande benefício à sociedade, esses sistemas muitas vezes também tem o poder de prejudicar quando neles ocorrem falhas: sejam danos a seres vivos, ambientais ou até econômicos. Quando se trata de sistemas críticos, deseja-se ter a garantia que esses sistemas sejam suficientemente seguros. Na busca de certificações de segurança para sistemas eletrônicos, destaque-se a norma IEC61508.*

No decorrer deste trabalho são vistos conceitos relacionados a segurança, e um estudo da norma IEC61508 na sua versão mais atual, 2.0 de 2010. Será dado maior enfoque na terceira parte da norma, referente ao desenvolmineto de software seguro. Ao final, é proposta uma ferramenta digital para o auxílio aos projetistas de software no complexo processo de certificação, como a norma em questão.

1. Introdução

1.1. Histórico de Segurança

A partir da década de 80, houve o crescimento na utilização de sistemas relacionados à segurança, em campos onde a garantia de funcionamento é um fator crítico, como a aviação, usinas nucleares, equipamentos médicos, ferrovias, aplicações militares e indústria. Juntamente a essa ascensão, houve uma série de iniciativas ao redor do mundo na criação de padrões e guias que permitissem o desenvolvimento desses sistemas, para que fossem comprovadamente seguros. Dentro dessas iniciativas, uma de grande destaque é a norma IEC61508, que é abordada no presente trabalho. [3], [6], [4]

1.2.Exemplos de Uso

Podemos ilustrar alguns sistemas onde a propriedade de *safety* é necessária, e a busca por certificações é vista como essencial:

- Em automóveis: nos sistemas de tração e freios ABS;
- Em maquinários e equipamentos na indústria ou indústrias de processos químicos danosos: sistemas de parada emergencial;
- Em hospitais: aparelhos de suporte a vida;

- Equipamentos utilizados em radioterapia: mecanismos de controle de dosagem de exposição e bloqueio;
- Em aeronaves: sistemas de controle de voo e orientadores de mísseis;
- Em ferrovias: mecanismos de sinalização;
- Em usinas nucleares: sistemas de desligamento emergencial [4]

1.3.Certificação de Software

Validação e verificação de qualidade de *hardware*, tanto para uso em segurança ou não, estão hoje no seu estado da arte. As medidas dos seus atributos são precisas; os limites de aceitação são muito bem definidos e o processo de avaliação utilizado hoje é tido, quase que universalmente, como ideal. É possível avaliar esses componentes não só quanto ao seu processo de fabricação, mas também como produto final.

Quando falamos em qualidade de software, muitos autores acreditam que há espaço para melhorias. Os padrões para o desenvolvimento de software seguro normalmente recomendam ou exigem certas técnicas a fim de reduzir os riscos que poderiam vir a causar acidentes ou danos. Essas recomendações normalmente são divididas em níveis de rigor, onde as mais exigentes são aplicadas nos casos onde as consequências de falhas são mais severas [5]. Apesar de alguns autores acreditarem que somente um processo certificado não garante de fato um produto com as propriedades de *safety*, essa abordagem apesar de não ser ideal é amplamente aplicada e usada comercialmente pois, na prática, é o que tem se mostrado factível. Existem interesse e espaço na melhoria de todo o processo de garantir software seguro, para que no futuro se tenha regras tão sólidas e de aceitação comum como em outras áreas do conhecimento, como é o caso da engenharia civil [4]. No escopo desse trabalho, iremos nos ater ao que se tem de mais sofisticado comercialmente hoje em dia, mas o autor acredita que essa discussão seja bastante válida.

1.3.1. Documentação

Um ponto de extrema importância nos processos de certificação de *software*, é a atenção dada à documentação durante todo desenvolvimento dos sistemas relativos à segurança. Ela permite aos avaliadores identificarem as precauções adotadas pelos desenvolvedores para atender os requisitos de segurança. Para atender normas como a IEC 61508, que possui um grande nível de complexidade e uma quantidade imensa de requisitos, é bastante comum o uso de ferramentas digitais que auxiliem o gerenciamento desses artefatos [02].

1.4.IEC

A *International Electrotechnical Commission* (IEC) é, segundo seu próprio site [1], uma instituição líder mundial na preparação e divulgação de padrões internacionais de tecnologias E/E/EP. É baseada em cooperação internacional, e é parte da tríplice de organizações (IEC, ISO, ITU) responsável pela criação de padrões internacionais.

1.5.IEC61508

A partir de 1985, a IEC começou uma iniciativa para o desenvolvimento de um padrão genérico para ser utilizado em sistemas E/E/EP relacionados à *safety*, diferenciando das abordagens de setores específicos na época (como aviação ou uso em usinas nucleares). Essa iniciativa deu origem, somente em 2000, a primeira edição da norma IEC 61508, dividida em 7 partes. Em 2005 foi publicado um adendo, IEC61508-0, meramente informativo, servindo de relatório técnico. A partir de 2002 se iniciou um processo de revisão, que teve sua edição 2.0 publicada em abril de 2010, na qual o presente trabalho se baseia. [3], [6].

IEC61508 tem se mostrado de grande sucesso e se destaca por abordar todos os componentes dos sistemas relacionados à segurança, bem como todas as etapas do processo, do planejamento e desenvolvimento até o desligamento e desativação desses sistemas. [6]

1.5.1. IEC61508-3 Requisitos de Software

A terceira parte da norma é relativa aos requisitos de software. Bastante extensa, incluindo seus anexos possui mais de 100 páginas. Nela são apresentados ciclos de vida, tanto de segurança funcional do sistema como um todo, e em detalhe o ciclo de vida de segurança de software. De acordo com cada etapa, são apresentadas as técnicas e medidas que devem ou não ser utilizadas, de acordo com cada nível de SIL. Essas técnicas são classificadas da seguinte maneira:

- HR: Altamente Recomendada. Caso não seja utilizada, deve ser propriamente justificada e aceita pelo órgão avaliador.
- R: Recomendada. Não é obrigatória, porém é sugerida.
- NR: Não Recomendada. Sua utilização implica também na necessidade de uma justificativa a ser aceita.
- --: Indiferente. Não há recomendação nem a favor, nem contrária a sua utilização.

2. Definições de Segurança

A fim de proporcionar uma melhor compreensão do presente trabalho, iremos rever a definição de alguns termos relacionados à segurança. Essa terminologia também é apresentada em língua inglesa.

2.1.Hazard

Qualquer condição que possa causar danos, ferimentos, ou morte à seres vivos; dano ou perda de sistemas, equipamentos ou propriedade; ou danos ao ambiente.

Um elemento que possua o potencial para causar um *hazard* necessitará de um *mecanismo de iniciação* para ocorrer. Este por sua vez é a seqüência de eventos que leva ao *hazard* acontecer. *Hazards* são a base para a definição de requisito de *safety*. [7]

2.2.Risco (Risk)

No processo de desenvolvimento de um *Safety-related System*, cada *hazard* é avaliado em termos do risco que ele oferece. Risco pode ser definido como a combinação da

probabilidade de ocorrer determinado dano e a severidade do mesmo, caso ocorra. A possibilidade pode ser classificada qualitativamente em: freqüente, provável, ocasional, remota, improvável e incrível. O nível de dano causado também pode ser classificado como: catastrófico, crítico, marginal ou negligenciável. [7]

2.3.Safety

Safety pode ser definido com a ausência de risco inaceitável de gerar um *hazard*[3], ou a segurança de funcionamento em situações críticas [8].

2.4.Segurança Funcional (Functional Safety)

Segurança funcional é a capacidade de um sistema ou equipamento operar corretamente de acordo com as suas entradas, dentro do contexto de *safety*. Deve-se considerar o sistema como um todo, inclusive sua interação com o ambiente.[3]

2.5.Função de Segurança (Safety Function)

Safety Function é a função implementada em um Sistema Relacionado à *Safety* que é responsável por alcançar ou manter um estado seguro do equipamento no evento de um potencial *hazard* específico [3]. No contexto da norma IEC 61508, cada *safety function* é definida com um grau de integridade: o *safety integrity level (SIL)* [7].

2.6.Safety Integrity Level (SIL)

O Nível de Integridade de Segurança (*SIL*) determina o grau de desempenho de segurança. Na norma em questão, varia em 4 níveis, sendo SIL4 o mais exigente, e SIL1 o de menor grau [3].

2.7.Sistema Relacionado à Safety (Safety-related System)

Safety-related System é um sistema que possui uma ou mais Funções de Segurança e é capaz de utilizá-las quando necessário, de acordo com um Nível de Integridade de Segurança ao qual ele foi projetado, a fim de evitar o acontecimento de *hazards* previstos para esse sistema. [3]

2.8.Software Criticality

A análise de *software criticality* permite diminuir o grau de exigência necessário para se alcançar um SIL, e, conseqüentemente diminui o esforço necessário para se demonstrar *safety*. No contexto da IEC 61508-3, a *criticality* possui 4 níveis (de C0 a C3) que, cruzadas com os 4 níveis de SIL impõe condições e exigências necessários para cada caso. Esta abordagem se limita, entretanto, a unidades de software com funcionalidades e interfaces restritas, como bibliotecas e *drivers* para sistemas operacionais. [6]

2.9.Safety Case

Safety Case é uma estrutura na qual a equipe de desenvolvimento faz declarações de como os requisitos de *safety* foram atingidos no produto em questão, justificando as medidas e

técnicas tomadas. A criação desses documentos que compõe o *safety case* podem complementar o processo de desenvolvimento de sistemas seguros. [7]

Existem abordagens para certificação que levam em consideração somente este artefato, ao invés da avaliação de todas as etapas do projeto. Essas abordagens permitiriam mais liberdade ao desenvolvimento, mas não são bem vistas quando tratam de *Safety-related Systems*, uma vez que gera uma grande dificuldade na avaliação pelas entidades certificadoras. [4]

3. Norma IEC61508

3.1. Objetivos

A norma IEC 61508 foi criada para atender as necessidades relacionadas à *safety* em sistemas nos quais as falhas podem causar danos não só a seres vivos, bem como equipamentos ou até o meio ambiente. Seu enfoque é ser um padrão genérico, porém possui complementos para a cobertura de inúmeras áreas específicas.

Sua metodologia visa [2]:

- Fornecer uma abordagem baseada em análise de riscos para definir e estruturar as exigências de desempenho de Sistemas Relacionados a *Safety*.

- Incentivar o crescimento de tecnologias relacionados a segurança, apresentando um padrão genérico, e permitindo sua especialização para setores específicos.

- Através de técnicas bem fundamentadas, fazer uso de uma abordagem sistemática e que seja flexível para adaptações futuras.

- Obter um aumento de confiabilidade, tanto de operadores como usuários em geral, em sistemas computadorizados com certificação em *safety*.

- Facilitar a criação de métodos para a avaliação e verificação de conformidade a requisitos de *safety*.

- Baseando-se em conceitos genéricos, definir requisitos que permitam a criação de uma rede eficiente de fornecimento de componentes a *Safety-related Systems* em diversas áreas.

- Permite o uso de diferentes graus de exigência, para a cobertura de 4 Níveis de Integridade de Segurança (*SIL*).

- Abordagem diferenciada para os modos de operação dos sistemas em *Low* e *High Demand*.

3.2. Estrutura

A norma IEC 61508 é dividida em 7 partes:

- 0 Functional safety and IEC 61508
- 1 General requirements
- 2 Requirements for electrical/electronic/programmable electronic safety-related systems
- 3 Software requirements
- 4 Definitions and abbreviations
- 5 Examples of methods for the determination of safety integrity levels
- 6 Guidelines on the application of parts 2 and 3

7 Overview of techniques and measures

As partes 1, 2 e 3 contêm todas as exigências da norma. Juntamente com a parte 4, compõe o set básico das publicações de *safety*. As partes 0, 5, 6, e 7 são complementares, e não contêm exigências normativas.

3.2.1. Publicações para Setores Específicos

São dignas de nota algumas publicações que são especializações da norma, e estão em conformidade com a IEC 61508:

IEC 62061: Utilizada para Maquinário

IEC 61511: Especializada em processos

IEC 61513: Direcionada a uso em Energia Nuclear

Esses padrões são baseados nos princípios de criação de subsistemas contidos na IEC 61508, e levam em consideração as práticas desses setores, bem como sua terminologia e medidas específicas [3].

3.3. Edição revisada 2.0 de 2010

O processo de revisão da norma IEC61508 começou em 2002 estendendo-se até 2010. Nesse processo são seguidas as seguintes etapas [3]:

1. Requisição da visão dos Comitês Nacionais quanto à norma.
2. Baseado nas opiniões do item 1, é preparado um esboço (*Comitee Draft*) para ser avaliado pelos Comitês Nacionais.
3. A partir da avaliação dos comitês é preparado um esboço para votação (*Committee Draft for Vote*)
4. Se for aceito, é preparada uma versão final das mudanças (*Final Committee Draft International Standard*). Elas somente serão publicadas se, novamente, for aprovada em votação.

Um fator com peso considerável na revisão é se, de fato as alterações agregam valor a norma, tendo em consideração também o impacto econômico dessas mudanças. No decorrer deste capítulo veremos algumas dessas mudanças aplicadas nesta nova edição.

3.3.1. Terminologia

Houve mudanças em algumas definições utilizadas na norma, com impacto direto na leitura e interpretação desses termos. Como exemplos nós temos a adição de *subsystem*, que possuía utilização inconsistente; *dangerous failure*; *safe failure*, *element* e *element safety function*.

3.3.2. Modos de Operação

Foram modificados os critérios que classificavam a operação de *high* e *low demand* de *safety function*: foram retiradas as exigências de *proof test frequency*.

3.3.3. Security

Agora as ações intencionais malélicas e não autorizadas devem ser consideradas na fase de análise de Riscos e *Hazards*. Se esses ataques forem previsíveis, medidas de segurança deverão ser tomadas. Essa medida mostrou-se importante para entrar em conformidade com outros padrões, tanto da IEC como da ISO.

3.3.4. Especificação de Exigências de E/E/PE

O processo de foi dividido em duas etapas, necessitando agora duas especificações:

1. Desenvolvimento da especificação de *E/E/PE system safety requirements* na parte 1 da norma.
2. Desenvolvimento da especificação de *E/E/PE system design requirements* na parte 2 da norma.

3.3.5. Comunicações Digitais

Nas arquiteturas de comunicação tipo *White Channel*, todo o canal de comunicação (inclusive protocolo, serviços e rede) deve estar em conformidade com as normas IEC61508, IEC 61784-3 ou IEC 62280. Nas do tipo *Black Channel*, a conformidade das interfaces deve ser com as normas IEC 61784-3 ou IEC 62280.

3.3.6. Gerenciamento de Segurança Funcional

Houve uma reestruturação nesse quesito, e agora é mandatória a nomeação de responsáveis para cada fase, bem como a identificação de todos os envolvidos em determinadas atividades relacionadas diretamente à busca de *safety* no desenvolvimento dos sistemas.

3.3.7. Circuitos Integrados e ASICS

Novas exigências foram incluídas, sendo adicionadas novas técnicas. 2 anexos à norma foram complementados, contemplando as mudanças necessárias.

3.3.8. Manual de Safety para Itens em Conformidade

Nessa revisão foi adicionado o conjunto de exigências para os fornecedores que alegam que seus produtos estão em conformidade com a norma. Esse manual deve conter toda a informação para que esse item em conformidade possa ser integrado com um Sistema Relacionado à *Safety*.

3.3.9. Software IEC 61508-3

Podemos listar as seguintes mudanças na parte 3 da norma, relativa à software:

Como saídas de cada fase do ciclo de vida, foram adicionadas propriedades desejadas, como: completude, corretude e previsibilidade.

Mecanismos para a reutilização de elementos de software que originalmente não foram desenvolvidos para *safety*, através da demonstração de segurança em outras aplicações.

As técnicas utilizadas no desenvolvimento de software presente nos Anexos A e B da norma foram revistos, removendo as entradas obsoletas ou de uso incomum, sendo substituídas por métodos mais comuns atualmente.

3.4.Considerações sobre a norma

A norma IEC 61508 é tida como o melhor padrão genérico para desenvolvimento de Sistemas Relacionados à *Safety*[7], se destacando por abordar todos os componentes destes, bem como todas as etapas do processo: do planejamento e desenvolvimento até o desligamento e desativação desses sistemas [6]. Sua abordagem é extremamente exigente nos seus requisitos. Possui uma análise de riscos muito aprofundada, o que na prática acaba levando a uma redução de custo total do processo de desenvolvimento e certificação para as companhias. Esse nível de exigência também beneficia a relação entre desenvolvedores e clientes: devido a especificações mais precisas, o entendimento entre as duas partes é facilitado[6].

Quando comparado a outros padrões que abordam o mesmo assunto, IEC 61508 tem vantagem ao ser ser mais prescritiva e focada na qualidade do produto; porém é ainda atrasado se considerarmos códigos de outros campos da engenharia, como a engenharia civil[4].

Uma liberdade oferecida pela norma é a utilização de técnicas não recomendadas (*NR*), ou ainda a não utilização de técnicas apontadas com altamente recomendadas (*HR*) para o desenvolvimento de software. Nesses casos, é necessária uma justificativa para tais atitudes: porém há dificuldade em determinar quais justificativas são aceitas pela entidade avaliadora, em cada caso. [2]

Como desvantagens, destaque-se o volume monstruoso da norma e a dificuldade na sua leitura e interpretação [6], por ser bastante textual, nem sempre expressa de maneira precisa e estruturada. Também apresenta deficiências para aplicação em alguns contextos específicos[7]. A complexidade e o alto custo do processo de certificação acaba afastando empresas de menor porte a buscar certificações em *safety*[6].

Outro ponto muitas vezes criticado diz respeito quanto ao emprego dos níveis de integridade de segurança. A complexidade que envolve os sistemas muitas vezes dificulta ou praticamente impossibilita uma estimativa precisa de SIL. Alia-se a isso o fato de que a interpretação do SIL apresenta divergências entre normas, além dos cálculos estimativos serem feitos baseados em considerações sobre confiabilidade[2].

Alguns autores [5] acreditam que, apesar da norma possuir uma série de orientações e servir de guia, ela acaba sendo mais focada na qualidade e repetibilidade do processo, do que, de fato em garantir propriedades de *safety*.

3.5.Alcançando Segurança Funcional

A estratégia para o alcance de segurança funcional tem como base [3]:

- Gerencimaneto de Segurança Funcional
- Exigências técnicas para as fases dos ciclos de vida relacionados à *safety*
- Avaliação de Segurança Funcional
- Nível de competência de indivíduos das equipes de desenvolvimento.

A norma utiliza uma abordagem baseada na análise de riscos, e para se alcançar um determinado Nível de Integridade de *Safety*, cada Função de Segurança deve ser tal que:

- A frequência de falhas dos Sistemas Relacionados à *Safety* é suficientemente baixa para que a frequência de eventos que possam causar *hazard* esteja de acordo com o nível de risco e/ou
- O Sistema Relacionado à *Safety* possa modificar as consequências do evento que possa causar *hazard* até um nível de risco tolerável.

É levado em consideração o modo de operação no qual o sistema esteja funcionando. Ele é classificado da seguinte maneira [3][7]:

- Baixa demanda: quando a demanda por operação da Função de Segurança não ultrapassa 1 por ano, ou maior que a frequência de testes. Tem sua medida na possibilidade de falha quando ativada.
- Alta demanda ou modo contínuo: quando a ativação é mais comum, tendo sua medida baseada na taxa de *hazards* por hora de funcionamento.

3.5.1. Ciclos de Vida

Estudos mostram [3] que a abordagem na criação de *safety-related systems* deve cobrir todas as fases de desenvolvimento. Isso se deve ao fato de diversas etapas mostrarem ter parcelas significativas na origem das falhas nos sistemas de controle. A figura 3.5.1 demonstra essa distribuição:

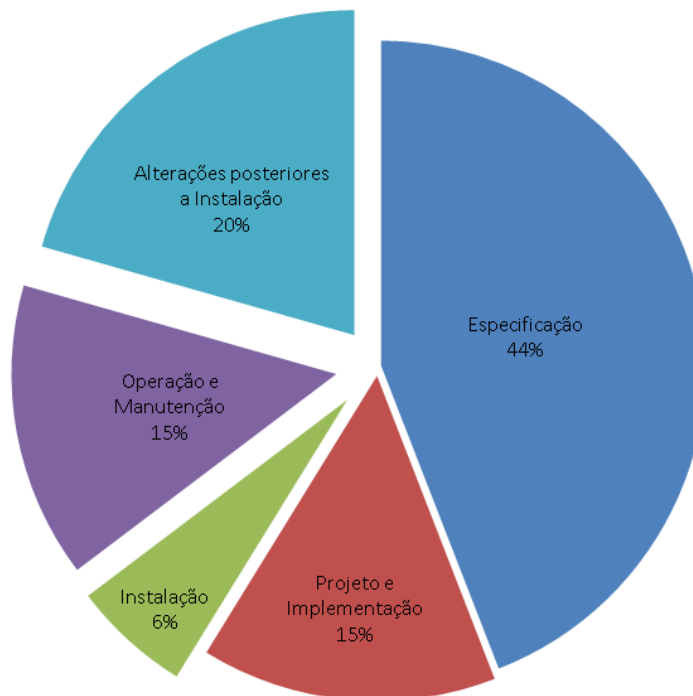


Figura 3.5.1: Origem das causas primárias de falhas em sistemas de controle [3]

É possível perceber que a maioria das falhas tem sido incorporadas aos sistemas durante o seu desenvolvimento, durante as fases de Especificação e Projeto e Implementação.

Com a finalidade de fornecer uma cobertura total, a norma IEC61508 apresenta 3 ciclos de vida importantes para lidar com Sistemas relacionados à *Safety*. Elas serão detalhadas a seguir:

- Ciclo de Vida de Segurança Funcional no capítulo [3.5.2](#)

Ciclo de Vida de Safety de Sistemas E/E/PE não é detalhado neste trabalho, e se refere à IEC 61508-2

- Software safety lifecycle no capítulo [3.6.3](#)

3.5.2. Ciclo de Vida de Segurança Funcional

Para lidar de maneira sistemática com as atividades necessárias para se alcançar um nível de *safety* almejado, a norma adota um Ciclo de Vida de Segurança Funcional, visto na Figura 3.5.2. Este é iniciado com definição do conceito, definição geral do escopo. Depois é feita a análise de de *hazards* e riscos. Juntas, essas atividades determinam *o que deve ser feito* (derivação dos requisitos de *safety*) e o nível de *safety* que deve ser alcançado (SIL) [7]. As 16 fases serão vistas com um nível maior de detalhe no decorrer deste capítulo [2].

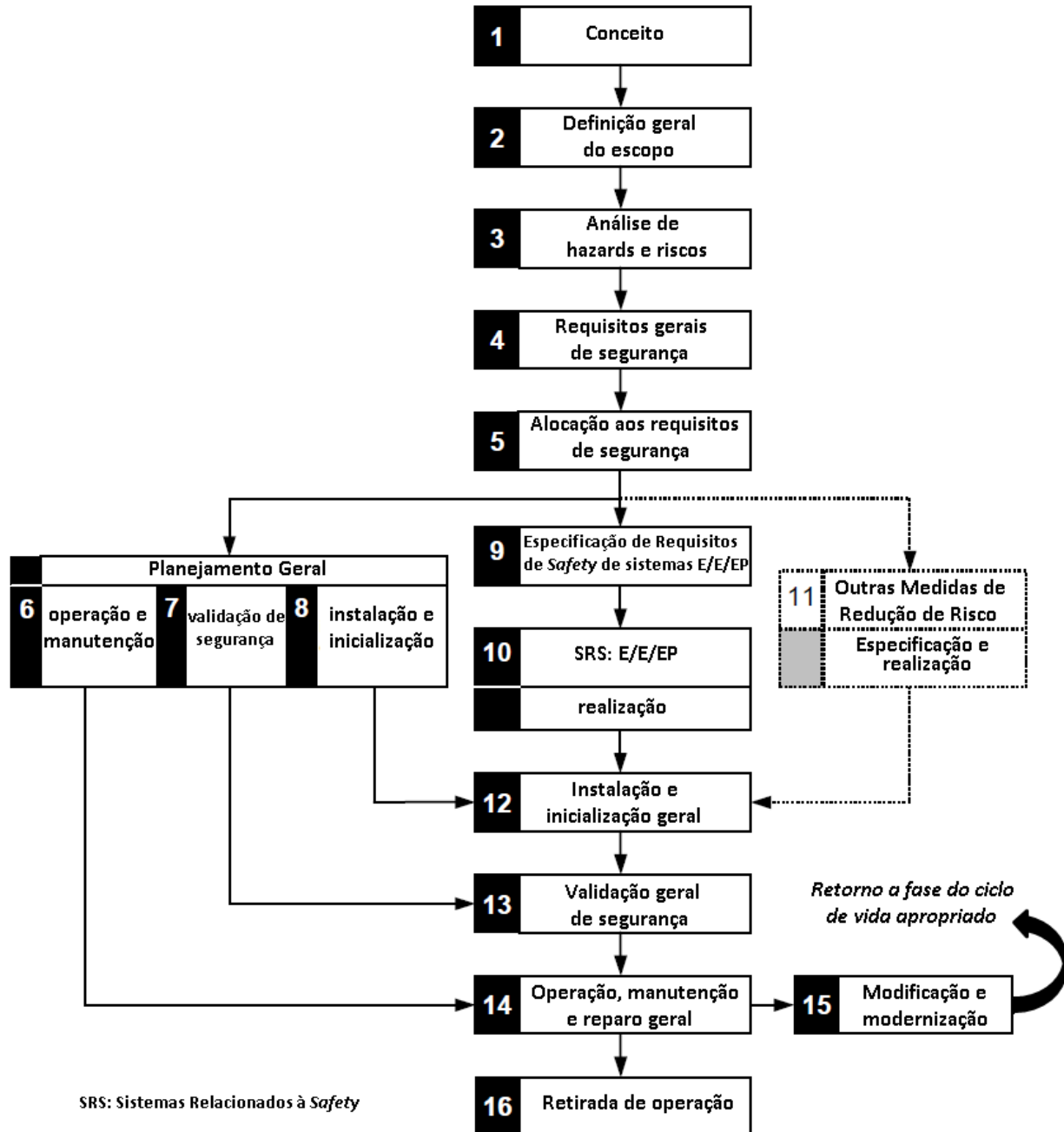


Figura 3.5.2: Ciclo de vida de Segurança Funcional.

3.5.2.1. Conceito

A fase inicial visa obter um entendimento geral do sistema e do ambiente relacionado. Deve ser satisfatório a ponto de possibilitar a execução correta das próximas fases do ciclo de vida.

3.5.2.2. Definição geral do escopo

A fase de definição de escopo tem por finalidade estabelecer a limitação do sistema de controle, bem como do sistema que estará sendo controlado. Auxilia na especificação do escopo das análises de *riscos* e de *hazards*.

3.5.2.3. Análise de *hazards* e riscos

A etapa de análise de *hazards* e riscos visa definir quais os riscos e eventos de *hazard* que possam ocorrer em determinado sistema. Isto se aplica em todas as situações passíveis de previsão e para os diversos modos de operação.

3.5.2.4. Requisitos gerais de segurança

Nesta parte do ciclo de vida devem ser especificados e desenvolvidos os requisitos gerais de segurança. Inclui os requisitos ligados à integridade de segurança e funções de segurança.

3.5.2.5. Alocação aos requisitos de segurança

Fase em que são feitas as alocações necessárias das funções de segurança, definidas nos requisitos gerais de segurança, para atingir o nível desejado de segurança funcional. As funções são alocadas aos sistemas de controle, sistemas externos de redução de *risco* e sistemas com outras tecnologias que estejam relacionadas à segurança.

3.5.2.6. Planejamento de operação e manutenção

Fase voltada ao planejamento dos processos de operação e manutenção de um sistema, visando assegurar a conservação das condições gerais de segurança ao longo destes processos.

3.5.2.7. Planejamento de validação de segurança

Esta etapa tem como finalidade a elaboração de uma estratégia que auxilie na aprovação da segurança do sistema como um todo.

3.5.2.8. Planejamento de instalação e inicialização

Esta fase tem como objetivo formular um plano para que tanto a instalação quanto a inicialização do sistema ocorram de forma controlada, para assegurar que sejam atingidas as metas de segurança funcional.

3.5.2.9. Especificação dos requisitos de *Safety* de sistemas E/E/EP

Durante essa fase são definidos os requisitos de segurança para os sistemas E/E/EP, de acordo com as exigências para se alcançar o Nível de Integridade de Segurança desejado.

3.5.2.10. Sistemas relacionados à *Safety* – Realização

A fase de realização tem por finalidade a elaboração do sistema responsável por efetuar o controle, de acordo com os requisitos levantados durante a fase apresentada em [3.5.2.9](#) tanto com relação à integridade de segurança, quanto para as funções de segurança.

3.5.2.11. Outras Medidas de Redução de Risco

Fase que tem como meta a inclusão de outras medidas e sistemas externos que atuem na redução de riscos (não pertence ao escopo da IEC 61508).

3.5.2.12. Instalação e inicialização geral

Fase onde é realizada a instalação e inicialização do sistema, seguindo os processos definidos anteriormente.

3.5.2.13. Validação geral de segurança

Nesta fase é executada a verificação do sistema de controle, para assegurar que esta satisfaça as especificações globais de segurança (nível de integridade de segurança e funções de segurança). Considera a alocação, anteriormente estabelecida, para os requisitos de segurança.

3.5.2.14. Operação, manutenção e reparo geral

Fase que tem por objetivo assegurar que a segurança funcional do sistema se mantenha durante os processos de operação, manutenção e reparos eventuais do sistema.

3.5.2.15. Modificação e modernização

Esta etapa tem como responsabilidade assegurar a manutenção da segurança funcional, seja antes ou depois de modificações feitas no sistema.

3.5.2.16. Retirada de operação

Etapa que tem como foco assegurar que as atribuições de segurança funcional sejam satisfatórias ao longo do processo de retirada de operação do sistema, bem como ao término deste processo.

3.6.EC61508-3

3.6.1. Software Safety

Software para Sistemas relacionados à *Safety* deve ser desenvolvido de tal maneira que satisfaça os requisitos de *safety* definidos para ele. Para permitir posteriormente a validação e verificação da entidade certificadora, é crucial que se mantenha o acompanhamento entre os requisitos de *safety* e as decisões tomadas no seu desenvolvimento [7].

3.6.2. Ciclo de vida de Software Safety

O processo de desenvolvimento de é organizado em etapas como na Figura 3.6.2. Dependendo do SIL almejado, a norma IEC61508 faz diferentes exigências, em cada uma dessas etapas. As fases podem conter subfases, e são detalhadas no decorrer deste capítulo. [2]

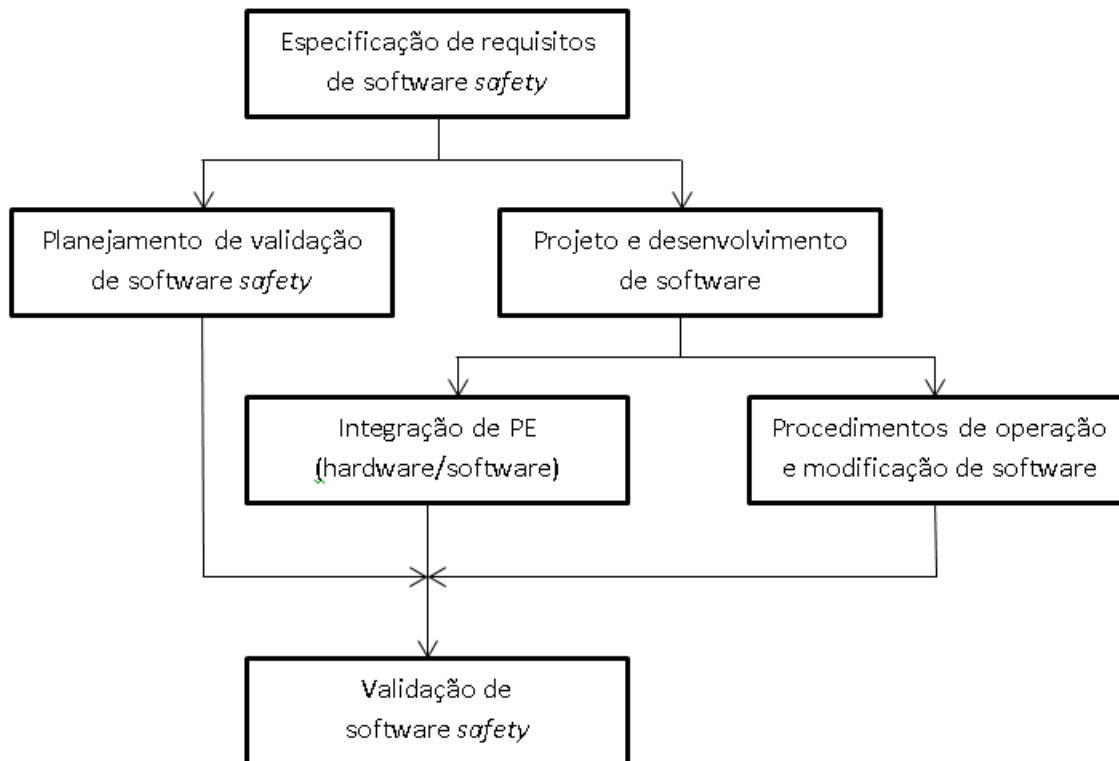


Figura 3.6.2: Ciclo de vida de software Safety

3.6.2.1. Especificação dos requisitos software safety

A especificação de requisitos de software *safety* deve abranger a definição de quais as Funções de Segurança serão executadas, assim como o modo de operação e quais os níveis de integridade de cada uma das Funções de Segurança.

3.6.2.2. Plano de validação de software safety

O plano de validação de software *safety* deve apresentar a estruturação do gerenciamento de *safety*, as atividades relativas à *safety* e também os marcos de aprovação presentes no ciclo de vida. Deve ser elaborado no início do ciclo de vida, sofrendo revisões caso sejam feitas mudanças no sistema inicial.

3.6.2.3. Projeto e desenvolvimento de software

Esta fase é dividida em subfases e será analisada em detalhe no capítulo [3.6.3](#).

3.6.2.4. Integração (hardware/software)

A etapa de integração visa comprovar que a interação entre hardware e software durante o desempenho de determinada função ocorre de forma correta. No caso de softwares com SIL superior a zero, é necessária a elaboração de um plano de teste de integração entre software e hardware no começo do ciclo de desenvolvimento.

3.6.2.5. Procedimentos de operação e manutenção de software

Esta fase tem como objetivo definir procedimentos para assegurar a conservação de segurança funcional ao longo dos processos de operação e manutenção do software. Devem-se estabelecer condutas a serem seguidas quando da ocorrência de falhas de software, bem como procedimentos para diagnosticar os defeitos. Igualmente, é necessário a definir processos para revalidação e quais os requisitos do relatório de manutenção.

3.6.2.6. Validação de segurança de software

Consiste em verificar se cada fase do ciclo satisfaz seus requisitos específicos de *safety*, identificado nas fases anteriores. Ou seja, é a validação do software contra sua especificação de requisitos de *safety*.

A análise de *safety* e os testes adequados devem ser executados e documentados, com as evidências de que o software cumpre os requisitos apresentadas em um documento de justificativa de *safety*.

3.6.3. Ciclo de projeto e desenvolvimento de Software

A fase de “Projeto e desenvolvimento de software” mencionado no capítulo anterior, pode ter as suas subfases visualizadas na Figura 3.6.3, representando o modelo de desenvolvimento em V. São aceitos outros modelos pela entidade, porém este é o mais usual.

Cada fase desse modelo é detalhada nos itens a seguir[2]:

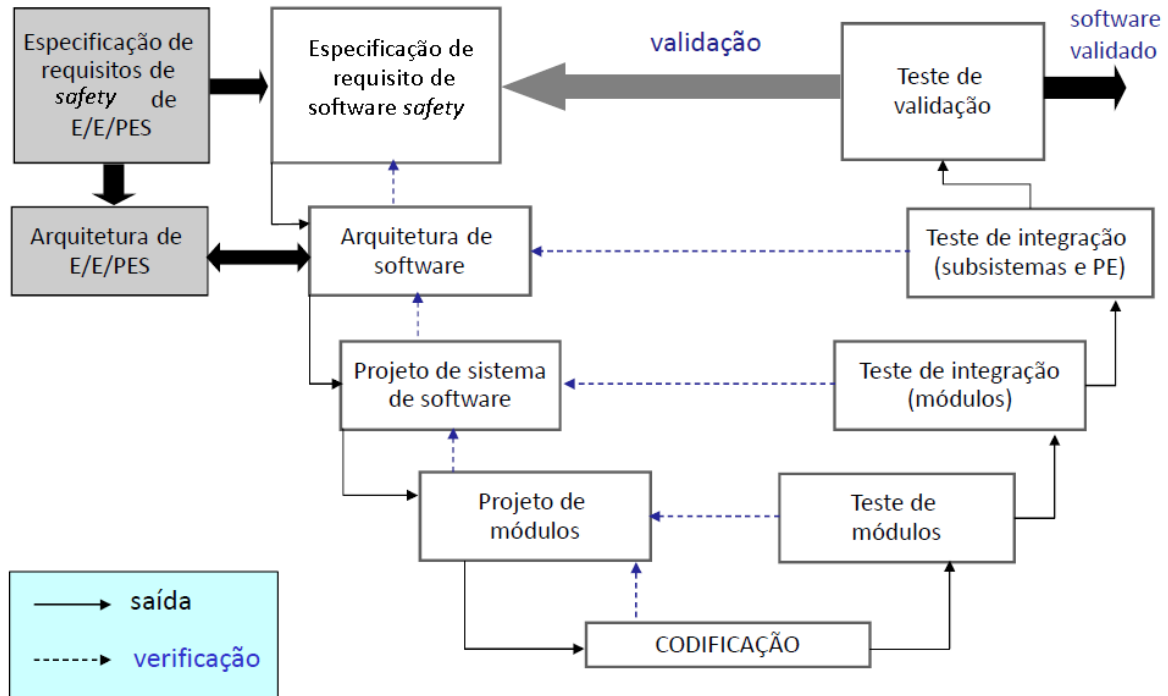


Figura 3.6.3: Modelo V de desenvolvimento de software. Adaptado de ([WEBER](#))

3.6.3.1. Arquitetura

A fase de arquitetura tem por objetivo elaborar a arquitetura de software visando cumprir os requisitos de *safety* definidos, de acordo com o nível de integridade de segurança solicitado. Também leva em consideração requisitos que tenham sido inseridos no software através da arquitetura do hardware e as interações software/hardware que possam ocorrer.

3.6.3.2. Suporte a ferramentas e linguagens de programação

O objetivo desta etapa é apurar um grupo de ferramentas que irão assessorar os processos de avaliação, validação, verificação e modificação do software. Estas ferramentas, dentre as quais estão os compiladores e linguagens de programação, devem ser apropriadas para o nível de integridade de segurança desejado.

3.6.3.3. Projeto de sistema

A meta desta etapa é elaborar o projeto de um software que seja passível de verificação, análise e que mantenha os atributos de *safety* durante processos de modificação. Além disso, deve satisfazer os requisitos de *safety* de software especificados para o nível de *Safety Integrity Level* exigido.

3.6.3.4. Projeto de módulos

Esta fase tem como finalidade a análise e divisão do software em módulos, refinando o projeto de arquitetura inicial de acordo com os subsistemas identificados. Da mesma forma, deve ser especificado o projeto de cada módulo individualmente, bem como os testes a serem efetuados em um determinado módulo do software.

3.6.3.5. Codificação

Ao longo da etapa de codificação, devem ser utilizadas normas que contenham boas práticas de codificação, especificando regras que restringem o uso de estruturas da linguagem, quando empregadas em módulos com requisitos de *safety*. Ainda, devem ser usadas ferramentas que detenham um certificado de validação com reconhecimento internacional.

A linguagem de programação deve ser compatível com os métodos de desenvolvimento e possuir mecanismos que auxiliem na identificação de incorreções no programa.

3.6.3.6. Teste de módulos

Esta etapa é responsável pelo teste de cada módulo do software, visando verificar se cada um destes executa as funções para as quais foram projetados e que satisfazem as definições exigidas de integridade e *safety functions*.

3.6.3.7. Teste de integração

Esta fase tem como finalidade verificar o cumprimento dos requisitos de segurança estabelecidos para o software. A validação é feita através de procedimentos que comprovem a interação correta e adequada entre os módulos, subsistemas e seus componentes. As atividades de teste de integração devem ser elaboradas concorrentemente ao longo das etapas de projeto e desenvolvimento.

3.6.4. Artefatos

Cada fase do ciclo de desenvolvimento gera um ou mais subprodutos. Estes artefatos gerados ao final de cada fase dos ciclos são utilizados como “matéria prima” para a fase posterior, ou seja, o *output* de uma fase é o *input* para a execução da próxima fase, formando uma cadeia. No contexto da IEC 61508, estes podem ser de variados formatos como por exemplo: especificações, descrições, diagramas, série de instruções, listas, *log* de atividades, plano, relatório [7]. O gerenciamento dessa série de itens é um desafio durante os projetos.

3.6.5. Backtracking

Durante a execução de projetos, é comum que algumas necessidades e detalhes do sistema desenvolvido só sejam percebidas em fases mais avançadas. Para se manter a conformidade e não haver impacto nos requisitos de *safety*, essas modificações do projeto inicial devem ser analisadas, e fases do ciclo de vida que já eram tidas como finalizadas devem ser revistas. Essa dinâmica é chamada de *backtracking*.

4. Implementação

O desenvolvimento de software voltado a certificações é um processo extenso e bastante complexo. Este capítulo aborda a proposta de criação de uma ferramenta digital que auxiliará equipes de desenvolvimento e gerenciamento de projetos relacionados à *safety*. Diferencia-se de outras caras ferramentas disponíveis no mercado por ser gratuita, sem fins comerciais.

4.1. Funcionalidades desejadas

De uma maneira geral, a ferramenta oferece uma plataforma para acompanhar o desenvolvimento de software. Durante cada fase de desenvolvimento, é possível fazer o controle das técnicas e medidas necessárias para se alcançar determinado SIL.

4.1.1. Pronta para o uso para certificação conforme IEC 61508-3 v2.0 de 2010

Ferramenta previamente configurada para o uso na norma, na sua versão mais atual.

4.1.2. Adaptabilidade

A ferramenta pode ser reconfigurada:

Para futuras atualizações na norma (como mudança nas técnicas).

Personalizável a outras normas, que possuam ciclos de vida similar à IEC61508.

4.1.3. Sistema de Acessos

Sistema de usuários, com permissões e acessos diferenciados, permitindo um maior controle de projeto.

4.1.4. Sistema de notificação para mudanças nas fases de desenvolvimento

Mudanças que impactam o projeto são notificadas aos usuários.

4.1.5. Perspectivas em Alto Nível e em Detalhe das Fases

Visualização em alto nível dos ciclos de vida do projeto, bem como a explosão para menor nível de granularidade. Desde a visão geral do projeto até a listagem de técnicas necessárias para qualquer subfase de desenvolvimento.

4.1.6. Suporte ao Controle de Documentação

Repositório para os artefatos gerados entre as fases de desenvolvimento.

4.1.7. Backtracking

Modificações que impactem fases passadas de projeto são sinalizadas, indicando quais fases devem ser revistas para que a conformidade à norma seja respeitada.

4.1.8. Multi Projetos

Mais de um projeto pode ser gerenciado pela ferramenta, através de um sistema de arquivos.

4.1.9. Logging

Quaisquer alterações realizadas no projeto são armazenadas, com data, usuário e tipo de alteração.

4.2. Desenvolvimento da Ferramenta

4.2.1. Cronograma

Na Tabela 1 temos o planejamento das atividades para o desenvolvimento da ferramenta em questão. O período referido se passa no segundo semestre de 2012.

| Atividade | Agosto | Setembro | Outubro | Novembro | Dezembro |
|---|--------|----------|---------|----------|----------|
| Escolha da linguagem de programação e ferramentas | X | | | | |
| Modelagem modular do sistema | X | X | | | |
| Desenvolvimento de Código | | X | X | | |
| Definição e execução dos testes | | | X | | |
| Análise dos resultados | | | X | X | |
| Escrita do trabalho final | | | X | X | |
| Apresentação | | | | | X |

Tabela 1: Cronograma de desenvolvimento

Bibliografia

1. <http://www.iec.ch> Acessado em 08/06/2012)
2. Bandeira, Diego C.: Aplicação da Norma IEC 61508 em Sistemas Críticos
3. Bell, Ronn: Introduction and Revision of IEC 61508
4. Wassying, A., Maibaum, T., Lawford, M., Bherer, H.: Software Certification: Is There a Case against Safety Cases?
5. McDermid, John A.: Software Safety: Where's the Evidence?
6. Faller, R.: Project Experience with IEC 61508 and Its Consequences
7. Panesar, Rajwinder K., Sabetzadeh, M., Briand, L.: Characterizing the Chain of Evidence for Software Safety Cases: A Conceptual Model Based on the IEC 61508 Standard
8. SIQUEIRA T. F.; MENEGOTTO, C.C.; WEBER, T. S.; NETTO, J.C.; WAGNER, F.R. Desenvolvimento de Sistemas Embarcados para Aplicações Críticas IV Escola Regional de Redes de Computadores, Passo Fundo, setembro 2006.
9. Weber, Taisy S.: Slides de Aula