CNMAI Congresso Nacional de Matemática Aplicada à Indústria 2014

18 a 21 de novembro de 2014, Caldas Novas - Goiás

FRAMEWORK PARA O DESENVOLVIMENTO DE PROGRAMAS DE CONTROLES DE SIS BASEADO NA NORMA IEC 61511

Rodrigo Cesar Ferrarezi, rferrarezi@usp.br[1]
Reinaldo Squillante Júnior, reinaldo.squillante@usp.br[1]
Jeferson A. L. Souza, jeferson.souza@usp.br[1]
José Reinaldo Silva, reinaldo@usp.br[1]
Fabrício Junqueira, fabri@usp.br[1]
Paulo Eigi Miyagi, pemiyagi@usp.br[1]
Lucas Antonio Moscato, lamoscat@usp.br[1]
Diolino J. Santos Filho, diolinos@usp.br[1]

[1]Escola Politécnica da Universidade de São Paulo – Departamento de Engenharia Mecatrônica e de Sistemas Mecânicos ... Av. Prof. Mello Moraes, 2231 - Cidade Universitária - São Paulo-SP - Brasil

Resumo: *Devido à alta complexidade dos sistemas produtivos atuais, o desenvolvimento de soluções de controle apropriadas de acordo com as normas industriais, os possíveis impactos negativos nos seres humanos e equipamentos e meio ambiente em caso de falha desses sistemas, existe uma demanda muito grande para o desenvolvimento de soluções de controle que sejam estáveis e seguros. Uma forma de se desenvolver sistemas mais seguros e confiáveis é o uso dos Sistemas Instrumentados de Segurança (SIS) de acordo com as normas IEC 61508 e IEC 61511. Entretanto, mesmo quando programas são desenvolvidos de acordo com as normas aplicáveis, programas de controle de SIS são susceptíveis de erros de desenvolvimento e especificação assim como no desenvolvimento de qualquer tipo de software. Uma forma de melhoria da confiabilidade desses programas de controle, que também é requerimento das normas IEC 61508 e IEC 61511, é a utilização do ciclo de desenvolvimento de programas de segurança de SIS em conjunto com técnicas de verificação formal dos modelos dos programas de controle, bem como uma abordagem unificada para a modelagem dos programas do SIS de prevenção e mitigação, dessa forma, possibilitando um melhor entendimento de suas interações.*

Palavras-chave: *IEC 61511, SIS de mitigação e prevenção, Model Checking, GHENeSys*

1. INTRODUÇÃO

Devido à crescente complexidade da automação aplicada aos sistemas produtivos, o desenvolvimento de soluções de controle para tais sistemas automatizados se torna uma tarefa cada vez mais complexa (Mazzolini et al., 2011). Além disso, as lógicas de controle que eram implementadas através de hardware, agora são implementadas através de softwares. Tal mudança permitiu que lógicas ainda mais complexas fossem implementadas (Bani Younis & Frey, 2003), e tais lógicas de controle necessitam de especificações mais detalhadas, assim como sistemáticas e metodologias para auxiliar seu desenvolvimento (Diaz, 2009).

Por mais inovador e moderno que um sistema de controle possa ser, os operadores, os equipamentos e o meio ambiente ainda podem ser colocados em risco caso falhas não possam ser devidamente tratadas (Sallak et al., 2008). Falhas ainda podem ocorrer mesmo nos sistemas mais modernos, pois, (i) nenhum dispositivo ou componente possui risco nulo de falhas, (ii) operadores humanos não possuem risco nulo e (iii) não é possível prever todos os estados atingíveis por um sistema (Squillante Júnior et al., 2011).

O conceito de Sistemas Instrumentados de Segurança (SIS) é uma possível solução para esses problemas, onde é aconselhável a implementação de camadas de redução de risco baseadas em soluções de controles hierarquicamente organizadas (Squillante Júnior et al., 2011). Tais soluções de controle são projetadas para gerenciar riscos através de processos de prevenção e mitigação, conduzindo o sistema a um estado seguro através de processos de degeneração (Squillante Júnior et al., 2011). Sistemas críticos – aqueles sistemas cujo risco aceitável é muito baixo, pois falhas podem causar acidentes graves – que são controlados por controladores programáveis (CLP) de segurança requerem que

suas lógicas sofram processos de verificação formal, bem como devem obedecer às normas de segurança (Wan et al., 2010). Uma forma de atingir esses requisitos é que se tenha o desenvolvimento dos programas de controle pautado pelo ciclo de desenvolvimento de programas de controle de segurança definido na norma IEC 61511 (Mayr et al., 2011).

No presente trabalho está sendo proposto um framework para auxiliar o desenvolvimento de programas de controle de SIS. O framework tem como principais características: (i) ser baseado nas fases do ciclo de desenvolvimento de programas de controle de segurança conforme as normas IEC 61508 (IEC, 2010) e IEC 61511 (IEC, 2003); (ii) o framework levará em consideração as relações e interações entre os programas do SIS de mitigação e de prevenção; (iii) será baseado na abordagem de desenvolvimento baseada em modelos (Mbd) (Mazzolini et al., 2011).

Este trabalho está organizado da seguinte forma: A seção 2 contém uma breve revisão dos principais conceitos utilizados neste trabalho, a seção 3 apresenta a descrição do framework proposto, a seção 4 apresenta a aplicação do framework em um caso de estudo real, e por fim, a seção 5 apresenta os principais resultados e conclusões finais.

2. PRINCIPAIS CONCEITOS

2.1. O Ambiente GHENeSys

Um sistema de controle SIS pode ser visto como um sistema dirigido por eventos que apresenta características funcionais como assincronismo, possibilidade de reset, paralelismos, concorrência, etc. e, dessa forma, essa classe de sistemas pode ser classificada como um sistema de eventos discretos, podendo ser modelada satisfatoriamente através de Redes de Petri e suas extensões (Murata, 1989), (Zurawski & Zhou , 1994).

O ambiente GHENeSys foi desenvolvido como uma extensão da Rede de Petri com orientação a objetos e mecanismos de abstração. Esses são definidos por meio de conceitos de hierarquia que estão inclusos, bem como mecanismos de síntese que são implementados através de uma abordagem estruturada que é suportada pelo encapsulamento introduzido pelo uso de objetos (del Foyo & Silva, 2003). O ambiente GHENeSys está sendo desenvolvido com o objetivo de representar de forma unificada Redes de Petri clássicas, suas extensões definidas na norma ISO/IEC 15909, bem como Redes de Petri de alto nível (del Foyo et al., 2011). O ambiente GHENeSys é composto do módulo de edição gráfica de redes GHENeSys, do módulo de simulação e do módulo verificador (del Foyo et al., 2011).

O ambiente GHENeSys também implementa diversos conceitos para facilitar os processos de modelagem, tais como: Pseudo-Boxes que permitem a modelagem da troca de informação entre diversas partes do sistema; Hierarquia que permite o encapsulamento de sub-redes sem perda de propriedades através do uso de macro elementos; Representação de intervalos de tempos não determinísticos, onde um intervalo de tempo pode ser definido para o disparo de uma transição ou permanência em um lugar.

A rede GHENeSys é a 7-upla $G = (L, A, F, K, \Pi, C_0, \tau)$, onde:

- $L = B \cup P$ é o conjunto de lugares que podem ser boxes ou pseudo-boxes.;
- A são as atividades, ou elementos ativos;
- $F \subseteq (L \times A \to \mathbb{N}) \cup (A \times L \to \mathbb{N})$ é a relação de fluxo;
- $K : L \to \mathbb{N}^+$ é a função capacidade;
- $\Pi : (B \cup A) \to \{0,1\}$ é a função que identifica macro elementos ou a hierarquia;
- $C_0 = \{(l, \sigma_j) | l \in L, \sigma_j \in \mathbb{R}^+ | l| < K(l)\}$ é o conjunto de marcações iniciais;
- $\tau : (B \cup A) \to \{\mathbb{Q}^+, \mathbb{Q}^+ \cup \{\infty\}\}$ é a função que mapeia os intervalos de tempo denso para cada elemento.

O conjunto de marcações é o par (l, σ_j) com $l \in L$, definindo que lugar cada marcação pode ser encontrado e σ_j definindo por quanto tempo essa marcação irá permanecer no lugar. O relógio é sincronizado globalmente e atualizado após cada transição.

O verificador do GHENeSys realiza a verificação formal de sistemas simultâneos de tempo real modelados através de redes GHENeSys pelo uso de técnicas de *Model Checking* (Clarke et al., 1999). O espaço de estados é construído através da abordagem enumerativa baseado no conceito de classe de estados (Berthomieu & Menasche, 1983). O verificador disponibiliza opções para construção de grafos tipo State Class Graph (SCG), Strong State Class Graph (SSCG) e Contracting the state class graph (CSCG). Propriedades a serem verificadas são especificadas através de lógica temporal Timed Computation Tree Logic (TCTL) (Alur et al., 1990).

O ambiente GHENeSys será utilizado neste trabalho devido: (i) Às características do SIS, pode haver uma grande quantidade de propriedades a serem checadas pelo verificador, e dessa forma é desejável que o espaço de estados seja gerado através da abordagem enumerativa ao invés de ser gerada "*on-the-fly*" repetidas vezes. Pois a geração do espaço de estados é feita em tempo e espaço exponencial, enquanto a verificação de uma propriedade é feita em tempo polinomial, caso o espaço de estados já esteja construído. (ii) À implementação da abordagem de tempo denso, pois várias propriedades do SIS podem ser dependentes do tempo. (iii) Ao formato Petri Net Markup Language (PNML) (ISO/IEC, 2005), que é implementado como formato de transferência padrão, permitindo o intercambio de informações entre os módulos do GHENeSys bem como com ferramentas externas que suportam o formato padrão, caso necessário. (iv) Ao ambiente, que permite que todas as atividades de modelagem e verificação sejam efetuadas sem necessidade de ferramentas externas. (v) À geração do espaço de estados e a especificação das propriedades a serem testadas é realizada na mesma ferramenta.

2.2. Metodologia PFS/MFG

2.2.1. *Production Flow Schema* (PFS)

O PFS foi criado para auxiliar o processo de modelagem de sistemas a eventos discretos (SEDs) baseando-se na ideia de que eles podem ser organizados de forma hierárquica. Isso é, um evento pode encapsular vários sub-eventos, permitindo uma abordagem *top-down* (Miyagi et al., 1988). De acordo com o PFS, qualquer processo produtivo pode ser modelado através de três elementos básicos: (i) Elementos atividade, que representam ações ou conjuntos de ações que transformam o item em processamento. (ii) Elementos distribuidores, que representam estados de espera e conectam uma ou mais atividades. (iii) Arcos, que representam relações entre os dois últimos elementos e a direção do fluxo de processamento. Na Figura 1 todos os elementos do PFS podem ser visualizados.

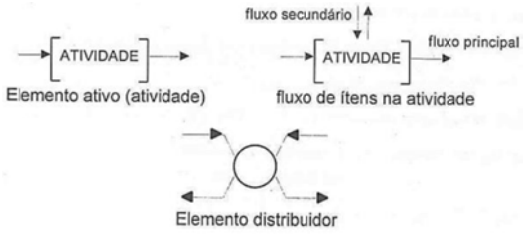

Figura 1. Elementos PFS

2.2.2. Metodologia PFS/MFG

Agora que os aspectos básicos do PFS foram definidos, a metodologia PFS/MFG para a construção de modelos pode ser descrita. O procedimento é composto por cinco passos (Miyagi, 1996):

- O processo principal deve ser definido e representado como um modelo PFS.
- O processo então deve ser detalhado em atividades ou recursos do sistema em desenvolvimento.
- As atividades e características são detalhadas em operações com a introdução de elementos Mark Flow Graph (MFG).
- Os elementos de controle são introduzidos e os recursos compartilhados são explicitados.
- Os sinais de controle de atuação e detecção são representados.

As redes MFG não serão detalhadas neste artigo. Os modelos MFG serão traduzidos diretamente para redes GHENeSys, pois ambos são grafos interpretados direcionados e bipartidos interpretados, que se originaram das Redes de Petri e portanto possuem mapeamento isomórfico.

3. DESCRIÇÃO DO FRAMEWORK

O framework proposto será desenvolvido de acordo com o ciclo de desenvolvimento de programas de segurança, também chamado de "modelo-V", da norma IEC 61511 (IEC, 2003). Na Figura 2 são demonstradas as relações entre cada componente do framework e as fases do "modelo-V". Cada componente do framework foi desenvolvido de forma a fornecer métodos e ferramentas para a implementação de cada fase do "modelo-V" e de acordo com a norma IEC 61511.

A primeira fase é a especificação dos requisitos do programa de segurança. Essa fase não faz parte do escopo do framework, porém deverá ser realizada pelas metodologias ou sistemáticas utilizadas para o desenvolvimento dos programas do SIS de prevenção e mitigação que serão discutidas a seguir.

Na primeira parte da segunda fase devem ser escolhidos os métodos, técnicas e ferramentas que serão utilizados no desenvolvimento dos programas de controle. Como já discutido na seção anterior, o ambiente GHENeSys será utilizado para modelagem e verificação dos programas de controle do SIS. O módulo verificador do ambiente GHENeSys requer que as propriedades sejam especificadas em lógica temporal TCTL, o qual permite que sejam especificadas propriedades importantes no contexto de SIS no domínio de tempo denso.

Além das ferramentas de modelagem, é necessário que se escolha uma metodologia de modelagem. Tal metodologia deve estar de acordo com os requisitos de modularidade exigidos pela norma IEC 61511, bem como permitir uma visão de alto nível do sistema em conjunto com os refinamentos requeridos. A metodologia PFS/MFG foi escolhida por atender aos requisitos delimitados anteriormente e por permitir a transformação isomórfica dos modelos em linguagens prescritas na norma IEC 61131-3 (IEC, 2003).

Como já mencionado, o desenvolvimento do framework será realizado conforme a abordagem baseada em modelos (MbD) (Mazzolini et al., 2011). Embora essa abordagem não seja prevista na norma IEC 61511, que cita que os programas de controle são diretamente desenvolvidos em uma linguagem de implementação. A norma requer que uma fase de validação final seja realizada no final do ciclo de desenvolvimento, e a modelagem é listada como uma das ferramentas recomendadas para validação. Dessa forma, ao se adotar o MbD, o framework não só estará em conformidade com a norma, mas também aprimorará a modularidade dos programas de controle SIS.

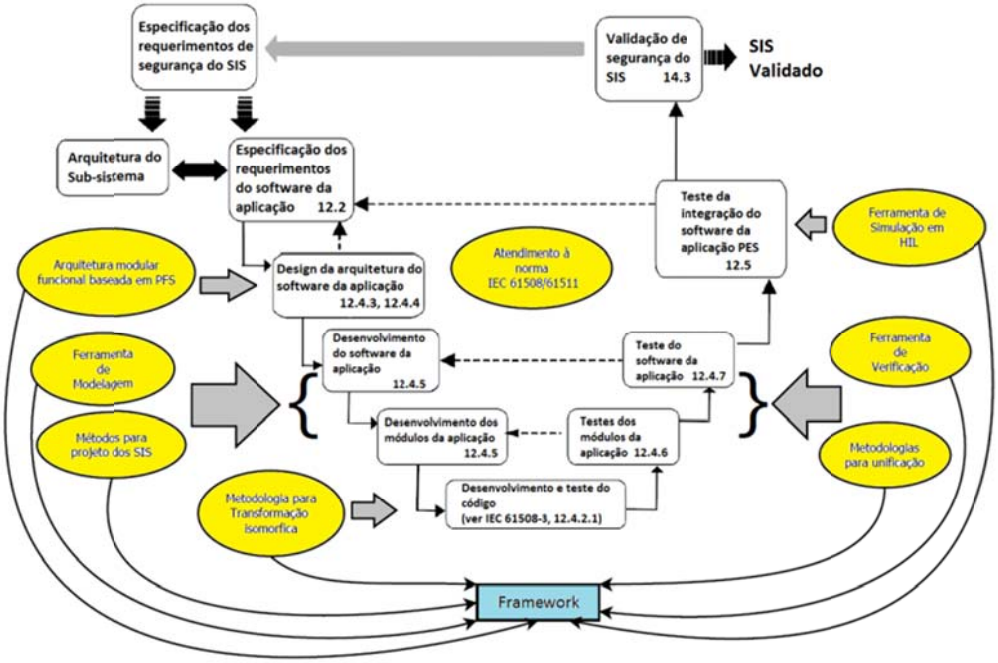

Figura 2. Framework proposto e relação com o "modelo-V"

3.1. Arquitetura do programa

Na segunda e última parte da segunda fase é proposta a arquitetura do programa de segurança. Através da aplicação da metodologia PFS/MFG, o nível mais alto de abstração foi definido como as atividades relacionadas com a prevenção e a mitigação de falhas críticas. Ambas as atividades podem então ser refinadas igualmente nas atividades de detecção e atuação. Na Figura 3 é demonstrado o modelo geral em PFS para o tratamento de uma falha após o refinamento das atividades.

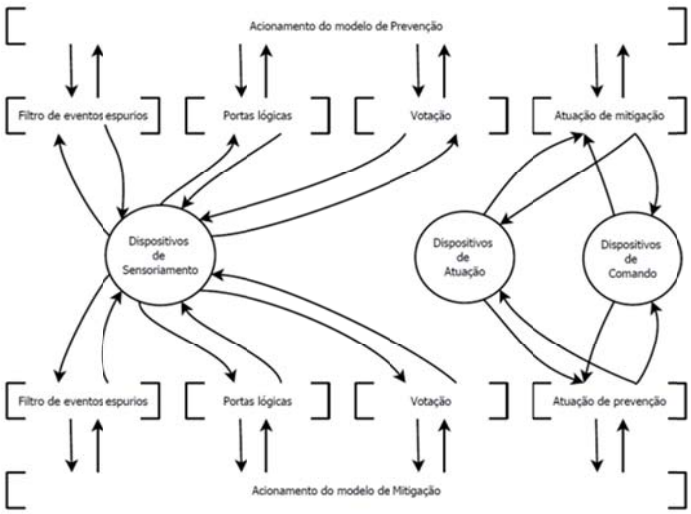

Figura 3. Modelo PFS do SIS

Refinando-se a atividade de detecção, temos as seguintes atividades: O filtro de eventos espúrios foi baseado nas ideias de (Squillante Júnior et al., 2011) e (Cavalheiro et al., 2011), e foi desenvolvido para evitar leituras espúrias dos sensores e, consequentemente, o acionamento indevido do SIS. As atividades de portas lógicas fornecem blocos lógicos "AND" e "OR" para implementação de lógicas de detecção entre sensores e lógicas de atuação entre atuadores. A atividade de votação fornece lógicas de votação como 1oo3, 2oo3, etc., pois elas são muito comuns no desenvolvimento de programas SIS. As atividades de alto nível apresentadas podem ou não ser implementadas de acordo com a especificação do SIS em desenvolvimento e podem trocar informações entre si, como por exemplo: três sensores de pressão podem ser ligados a um filtro de eventos espúrio cada, esses por sua vez são conectados a um bloco de votação

2oo3, e sendo esse último conectado então ao bloco de acionamento do modelo. O bloco de acionamento do modelo é responsável pela conclusão de que uma falha foi detectada a partir da informação recebida dos outros blocos e pela troca de informação com as atividades de atuação.

As atividades de atuação são responsáveis pela troca de informação com os atuadores e dispositivos de comando. Essas atividades implementam as ações necessárias para prevenir uma falta ou mitigar as consequências dessa falta. Embora não representado na Figura 3, atividades de portas lógicas podem ser utilizadas para implementar lógicas entre os atuadores ou sequenciamentos.

É importante notar que o framework apresentado aqui requer uma relação forte entre os SIS de prevenção e mitigação. Independente da metodologia aplicada para seu desenvolvimento, o SIS de mitigação deve ser projetado para mitigar as consequências do SIS como prevenção no caso de não ser efetivo ou mesmo falhar completamente em trazer o sistema para um estado seguro.

Seguindo os passos da metodologia PFS/MFG, os recursos e sinais de controle, tais como atuadores, sensores, sinais de comando e de temporização são adicionados às atividades resultando nos blocos. Na Figura 4 é demonstrado um exemplo do refinamento do bloco do filtro de eventos espúrios, onde suas restrições temporais podem ser vistas. Também é possível verificar como o bloco troca informação com o sensor e os blocos subjacentes – que podem ser blocos de votação, blocos lógicos ou de acionamento do modelo - através do uso de arcos habilitadores e inibidores.

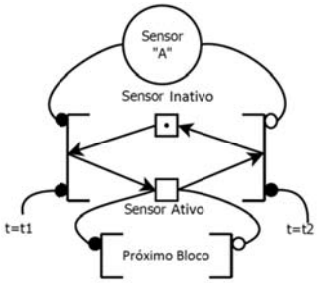

Figura 4. Filtro de eventos espúrios e conexões

3.2. Desenvolvimento do programa de controle

A terceira e quarta fase do ciclo de desenvolvimento está relacionada com o desenvolvimento do programa de controle. Nessa fase deverão ser escolhidas metodologias para o desenvolvimento das atividades de prevenção e mitigação. Tais metodologias devem ser baseadas em modelos e técnicas formais bem como estarem de acordo com os requerimentos das normas IEC 61508 e IEC 61511.

3.2.1. Metodologia para o desenvolvimento da atividade de prevenção

Primeiramente, a metodologia para o desenvolvimento da atividade de prevenção deve ser capaz de definir quais falhas serão tratadas pelo SIS. Tal definição pode ser feita através do estudo de hazard and operability (HAZOP), matriz causa-efeito ou outra técnica aplicável. Com o conhecimento das falhas, devem ser apresentados métodos formais para descoberta das relações causais que levam a detecção de cada falha, ou seja, qual sensor ou combinação de sensores é capaz de detectar determinada falha, e como esses sensores devem estar logicamente conectados. A metodologia então deve ser capaz de propor ações para o tratamento de cada falha através da atuação de algum componente, desligamento de um componente em perigo, entre outros. Além disso, caso necessário, devem ser definidos sinais de comando, e caso o filtro de eventos espúrios seja implementado, os tempos utilizados no bloco deverão ser informados.

Com as informações acima, o desenvolvimento do modelo PSF/MFG deverá ser continuado. Deverão ser selecionados os blocos necessários juntamente com a representação das trocas de informação entre eles. Com os blocos necessários conectados, os modelos podem então ser refinados em redes GHENeSys na ferramenta de modelagem.

3.2.2. Metodologia para o desenvolvimento da atividade de mitigação

Por sua vez, uma metodologia para o desenvolvimento da atividade de mitigação deve ser escolhida. A mesma metodologia já utilizada na prevenção, bem como outras metodologias, poderão ser escolhidas desde que os mesmos critérios descritos na seção anterior sejam empregados. Ao contrario da atividade de prevenção, a metodologia de mitigação não necessita definir as falhas a serem tratadas, pois o sistema de mitigação irá tratar as consequências do sistema de prevenção caso ele não seja capaz de tratar uma falha e trazer o sistema para um estado seguro. Assim como para a atividade de prevenção, com as informações necessárias os blocos podem ser conectados e os modelos podem então ser refinados em redes GHENeSys.

3.3. Processos de verificação

A sexta e sétima fases do ciclo de desenvolvimento estão relacionadas com a verificação formal dos modelos. A quinta fase será detalhada posteriormente. Devido a abordagem baseada em modelos, as verificações são realizadas nos modelos dos programas de controle e não no código de controle. Todas propriedades especificadas em linguagem natural serão traduzidas para expressões TCTL conforme os padrões propostos em (Dwyer et al., 1998). Os processos de verificação serão realizados em três diferentes níveis como detalhado abaixo.

3.3.1. Nível dos modelos

A primeira verificação é realizada em cada modelo de prevenção e mitigação de cada falha separadamente. Nesse nível é verificado se os modelos são capazes de identificar e tratar as falhas que eles foram projetados para identificar. A alcançabilidade de estados não desejados também deve ser verificada. As propriedades funcionais desejadas podem ser extraídas da descrição de cada SIF, e as propriedades não desejáveis podem ser retiradas da estrutura de cada modelo.

3.3.2. Nível do tratamento de uma falha

Com os eventuais erros nos modelos corrigidos, os modelos de prevenção e mitigação de uma falha devem ser integrados em um modelo de tratamento unificado. A integração dos modelos deve ser realizada conforme os seguintes passos:

- Primeiramente os sinais dos sensores compartilhados entre os modelos devem ser duplicados; cada modelo deve manter seu próprio bloco do filtro de eventos espúrios, seus blocos lógicos ou de votação. Somente o sinal do sensor deve ser compartilhado
- Os sinais dos atuadores devem ser primeiramente agrupados aos modelos que os compartilham. A atuação de atuadores compartilhados é implementada através blocos lógicos “OR” entre os blocos de acionamento pertinentes. Sequenciamentos de atuação devem ser implementados através de blocos lógicos “AND” entre os blocos de acionamento pertinentes.

O segundo nível busca verificar se os modelos integrados são capazes de trabalhar em conjunto, ou seja, se o modelo de prevenção pode prejudicar o funcionamento do modelo de mitigação e vice-versa. As propriedades em linguagem natural podem ser derivadas da estrutura do modelo integrado representado em redes GHENeSys.

3.3.3. Nível do SIS

O último passo de verificação é realizado no modelo completo do SIS, onde o tratamento de várias falhas está integrado. A integração dos modelos é realizada conforme procedimento descrito na seção anterior. O terceiro nível procura verificar se os modelos integrados de tratamento das falhas são capazes de trabalhar em conjunto de forma sinérgica, ou seja, se o tratamento de uma falha pode atrapalhar o tratamento de outras. As propriedades em linguagem natural podem ser derivadas da estrutura do modelo completo representado em redes GHENeSys.

3.4. Codificação do programa de controle

A quinta fase do ciclo de desenvolvimento é realizada no framework apresentado neste trabalho através de uma transformação isomórfica dos modelos em redes GHENeSys para uma linguagem prevista na norma IEC 61131-3. Para essa fase será utilizada a técnica proposta em (Thapa et al., 2005). Essa técnica permite a transformação de Redes de Petri, e por consequência redes GHENeSys, em diagramas LD.

3.5. Testes do programa de controle

A oitava fase do ciclo de desenvolvimento é o teste final no programa de controle. No framework aqui proposto, o programa de controle já traduzido para uma linguagem padrão da norma IEC 61131-3 deverá ser carregado para um PLC de segurança e submetido a testes exaustivos. Tais testes poderão ser realizados através de técnicas de Hardware-in-the-loop (HIL) (Gu et al., 2007). Essa fase deverá ser detalhada em trabalhos futuros.

4. APLICAÇÃO DO FRAMEWORK

O framework proposto foi aplicado ao desenvolvimento do programa de controle do SIS de uma estação de compressão de gás natural (ECOMP).

4.1. Descrição do processo

A estação de compressão ECOMP é conectada ao gasoduto por várias linhas de entrada, chamadas de sucção. O gás que entra na estação é filtrado em dois filtros coalescentes e então distribuído para quatro turbo compressores. O

gás já comprimido retorna ao gasoduto através das linhas de descarga. Existem sensores de temperatura, gás e fogo nas linhas de descarga de todos os compressores bem como sensores de pressão na linha de descarga principal. Várias válvulas ON-OFF estão instaladas nas linhas de sucção, nas linhas entre os filtros e os turbo compressores, e nas linhas de descarga. Existem também válvulas ON-OFF conectando às linhas de descarga de emergência e nas linhas dos cilindros de CO_2 conectadas a sucção de cada compressor. Todos TAGs de equipamentos usados nesse exemplo estão de acordo com a norma SA-S5.1-1984 (ISA, 1984 (R1992)).

4.2. Desenvolvimento do programa de controle

Agora com o processo descrito, as metodologias para o desenvolvimento das atividades de prevenção e mitigação devem ser selecionadas.

4.2.1. Desenvolvimento da atividade de prevenção

A atividade de prevenção será desenvolvida de acordo com a metodologia proposta em (Squillante Júnior et al., 2011). De forma breve, as falhas a serem tratadas são derivadas do estudo de HAZOP. Os modelos de detecção são gerados a partir de matrizes causa-efeito, onde essas são convertidas em redes Bayesianas e finalmente convertidas em Redes de Petri. Os modelos de tratamento, ou seja, os atuadores relacionados com o tratamento de cada falha são extraídos do relatório de HAZOP.

A aplicação da metodologia na ECOMP para aquisição das informações necessárias para o framework resulta em uma tabela contendo a identificação de cada SIF, a descrição de cada falha, os eventos inicializadores – os sensores que são capazes de detectar cada falha – e as ações de prevenção listando as válvulas que necessitam ser atuadas e os equipamentos que necessitam ser desligados. As relações lógicas entre os sensores relacionados com cada falha e os sinais de comando podem ser encontrados nas Redes de Petri resultantes.

A primeira falha a ser tratada é "Pressão alta nas linhas de descarga". Essa falha pode ser detectada pelos sensores $(\mathrm{PIT}-006\mathrm{A}, \mathrm{PIT}-006\mathrm{B}, \mathrm{PIT}-006\mathrm{C})$ em votação 2oo3. O tratamento será feito pelo fechamento de seis válvulas e pelo desligamento dos quatro compressores. A segunda falha a ser tratada é "Temperatura muito alta na descarga dos turbo compressores". Essa falha pode ser detectada pelos sensores em lógica na forma $OR(\mathrm{TIT}-209\mathrm{A}, A\mathrm{TIT}-209\mathrm{B}, \mathrm{TIT}-209\mathrm{C}, \mathrm{TIT}-209\mathrm{D})$. O tratamento será realizado pelo desligamento dos quatro turbo compressores. Um sinal de comando reiniciando os modelos também foi implementado.

Por fim, ambos os modelos de prevenção utilizarão o filtro de eventos espúrios, o bloco de acionamento do modelo e o bloco de atuação. A primeira falha ainda utilizará o bloco de votação e a segunda falha blocos de portas lógicas. Com os blocos necessários, os modelos PFS de alto nível são montados, refinados e convertidos em redes GHENeSys.

4.2.2. Desenvolvimento da atividade de mitigação

A atividade de mitigação será desenvolvida de acordo com a metodologia proposta em (Souza et al., 2013). O tratamento de mitigação tem como foco a mitigação dos efeitos de uma falha no tratamento de prevenção. Os modelos de detecção são gerados a partir de como detectar os efeitos da falha em um estudo de FMEA. As relações entre os sensores são descritas no estudo do FTA. Os modelos de tratamento são derivados de técnicas *What-if*.

Ao se aplicar a metodologia para a ECOMP, temos os estudos de FMEA, FTA e *What-if*. A partir deles é possível obter os sensores e atuadores relacionados com cada efeito, bem como suas relações lógicas e então construir os modelos da mesma forma que foi feito na atividade de prevenção.

Como exemplo, considere que fogo na planta seja um efeito da primeira falha de prevenção por não ter sido tratada devidamente. Esse efeito pode ser detectado pelos sensores em lógica $OR(\mathrm{BSH}-501, \mathrm{BSH}-502, \mathrm{BSH}-503, \mathrm{BSH}-504)$. O tratamento será inicialmente realizado através das mesmas ações previstas para essa falha na prevenção, após isso, deverá ser feita a abertura das válvulas da linha de descarga de emergência, fechamento das válvulas de saída dos filtros e somente depois, deverá ser realizada a abertura das válvulas dos tanques de CO_2.

Por fim, o tratamento do efeito demonstrado no exemplo utilizará o filtro de eventos espúrios, blocos de portas lógicas, o bloco de acionamento do modelo e o bloco de atuação. Blocos lógicos também serão necessários na atuação, pois alguns atuadores são compartilhados entre o tratamento de mais de um efeito. Com os blocos necessários, os modelos PFS de alto nível são montados, refinados e convertidos em redes GHENeSys.

4.3. Verificação dos modelos

Para cada nível de verificação foram propostos de três a seis propriedades, resultando em um total de 28 propriedades. Alguns exemplos de propriedades verificadas e suas respectivas proposições TCTL podem ser visualizados na Tabela 1.

Tabela 1. Propriedades verificadas

Nível dos Modelos – Falha 1P (prevenção)
Atuadores se mantem ativados e os compressores continuam desligados até que a falha não seja mais detectada pelos sensores por 10s e o sinal de comando esteja ativo.
$\forall \Big((\text{Actuators\_Active\_1P})\ \mathrm{U}_{\geq 10} \big((\text{not\_PSHH} - 006\text{A} \wedge \text{not\_PSHH} - 006\text{B}) \vee (\text{not\_PSHH} - 006\text{A} \wedge \text{not\_PSHH} - 006\text{C}) \vee (\text{not\_PSHH} - 006\text{B} \wedge \text{not\_PSHH} - 006\text{C}) \wedge \text{ENABLE\_EXT} \big) \Big)$
Nível do SIS – Integrado
A ativação de qualquer modelo de prevenção nunca ativa quaisquer atuador exclusivamente relacionado com os modelos de mitigação.
$\forall \square_{\geq 0} \Big(\text{Fault\_1P} \wedge \text{Fault\_4P} \wedge \neg \text{Fault\_1MA} \wedge \neg \text{Fault\_1MB} \wedge \neg \text{Fault\_1MC} \wedge \neg \text{Fault\_4MA} \wedge \neg \text{Fault\_4MC} \rightarrow \forall \square_{\geq 0} \big(\neg (\text{Actuator\_Active\_M} \wedge \text{Actuator\_Active\_M} - \text{I}) \big) \Big)$

Onde “P” quer dizer prevenção, “M” mitigação, “A”, “B” e “C” são os efeitos tratados pelos modelos de mitigação e “I” é para o efeito “incêndio”.

5. CONCLUSÕES

No presente trabalho foi proposto um framework para o desenvolvimento de programas de controle de SIS baseados na norma IEC 61511 associado à abordagem de desenvolvimento por modelos. Conforme requerido pela IEC 61511 foram escolhidos métodos e ferramentas para a execução de várias fases do ciclo de desenvolvimento, uma arquitetura para os programas de controle foi proposta, recomendações para escolha das metodologias para geração das atividades de prevenção e mitigação foram propostas e metodologias para integração dos modelos durante a verificação foram desenvolvidas.

O framework proposto foi então aplicado a um caso real de desenvolvimento de um SIS, resultando nos modelos dos programas de controle. Os modelos de prevenção tiveram resultados satisfatórios para todas as verificações. Os modelos de mitigação ainda estão em fase de geração do espaço de estados. Por fim, é esperado demonstrar aos engenheiros de controle que desenvolvem programas de controle de segurança para SIS a importância de seguir um framework desenvolvido de acordo com as normas IEC 61508 e IEC 61511, bem como que disponibilize métodos para que o projeto seja realizado em conformidade com as normas.

6. AGRADECIMENTOS

Os autores gostariam de agradecer as agências governamentais CNPq, FAPESP, MEC (PET - Programa de Educação Tutorial) e CAPES pelo apoio financeiro para a realização deste trabalho.

REFERÊNCIAS

Alur, R., Courcoubetis, C.A. & Dill, D.L., 1990. Model-checking for real-time systems. In *Proceedings of the Fifth Annual IEEE Symposium on Logic in Computer Science*. Philadelphia, 1990. pp.414-25.

Bani Younis, M. & Frey, G., 2003. Formalization of Existing PLC Programs: A survey. Kaiserslautern, 2003.

Berthomieu, B. & Menasche, M., 1983. An Enumerative Approach For Analyzing Time Petri Nets. In *Proceedings IFIP*. Paris, 1983. Elsevier Science Publishers. pp.41-46.

Cavalheiro, A.C.M. et al., 2011. Specification of Supervisory Control Systems for Ventricular Assist Devices. *Artificial Organs*, 35(5), pp.465–70.

Clarke, M., Grumberg, & Peled, D.A., 1999. *Model Cheking*. 1st ed. Cambridge: MIT Press.

del Foyo, P.M.G., Miralles, A.S.P.J. & Silva, J.R., 2011. UM VERIFICADOR FORMAL EFICIENTE PARA SISTEMAS DE TEMPO REAL. In *X SBAI – Simpósio Brasileiro de Automação Inteligente*. São João del-Rei, 2011. pp.1220-25.

del Foyo, P.M.G. & Silva, J.R., 2003. Towards a unified view of Petri nets and object oriented modeling. In *In 17th International Congress in Mechanical Engineering*. São Paulo, 2003. pp.518-24.

Diaz, M., 2009. *Petri Nets - Fundamental Models, Verification and Applications*. London: John Wiley & Sons.

Dwyer, M.B., Avrunin, G.S. & Corbett, J.C., 1998. Property Specification Patterns for Finite-state Verication. In *Proceedings of 2nd Workshop on Formal Methods in Software Practice*. Clearwater Beach, 1998. pp.7-15.

Gu, F., Harrison, W.S., Tilbury, D.M. & Yuan, C., 2007. Hardware-In-The-Loop for Manufacturing Automation Control: Current Status and Identified Needs. In *CASE 2007. IEEE International Conference on Automation Science and Engineering*. Scottsdale, 2007. pp.1105-10.

IEC, 2003. *IEC 61131-3 - Programmable controllers - Part 3: Programming languages*. Geneva: International Electrotechnical Commission.

IEC, 2003. *IEC 61511 - Safety instrumented systems for the process industry sector*. Geneva: International Electrotechnical Commission International Electrotechnical Commission.
IEC, 2010. *IEC 61508 - Functional safety of electrical/electronic/programmable electronic safety-related systems*. Geneva, Switzerland: International Electrotechnical Commission International Electrotechnical Commission.
ISA, 1984 (R1992). *ANSI/ISA-S5.1 — Instrumentation Symbols and Identification*. Research Triangle Park: Instrument Society of America.
ISO/IEC, 2005. *Software and Systems Engineering - High-level Petri Nets, Part 2: Transfer Format, International Standard WD ISO/IEC 15909. Wd version 0.9.0*.
Mayr, A., Plösch, R. & Saft, M., 2011. Towards an Operational Safety Standard for Software - Modelling IEC 61508 Part 3. In *18th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*. Las Vegas, 2011. IEEE Computer Society. pp.97-104.
Mazzolini, , Brusaferri, & Carpanzano, E., 2011. An Integrated Framework for Model-based Design and Verification of discrete Automation Solutions. In *Proceedings 2011 9th IEEE International Conference on Industrial Informatics*. Milan, 2011. pp.545-50.
Miyagi, P.E., 1996. *Controle Programável - Fundamentos do Controle de Sistemas*. São Paulo: Editora Edgard Blücher Ltda.
Miyagi, P.E., Hasegawa, K. & Takahashi, K., 1988. A programming language for discrete production systems based on Production Flow Schema and Mark Flow Graph. *Trans. SICE 24*, pp.183-90.
Murata, T., 1989. Petri nets: Properties, analysis and applications. *Proceedings of IEEE*, 77(4), pp.541–80.
Sallak, M., Simon, C. & Aubry, J.-F., 2008. A Fuzzy Probabilistic Approach for Determining Safety Integrity Level. *IEEE Transactions on Fuzzy Systems*, 16(1), pp.239-48.
Souza, J.A.L. et al., 2013. Mitigation Control of Faults in Critical Production Systems. In *International Congress of Mechanical Engineering*. Ribeirão Preto, 2013. pp.3889-99.
Squillante Júnior, R., Santos Filho, D.J., Junqueira, F. & Miyagi, P.E., 2011. Development of Control Systems for Safety Instrumented Systems. *IEEE Latin America Transactions*, 9(4), pp.451-57.
Thapa, D., Dangol, S. & Wang, G.-N., 2005. Transformation from Petri Nets Model to Programmable Logic Controller using One-to-One Mapping Technique. In *International Conference on Computational Intelligence for Modelling, Control and Automation*. Vienna, 2005. pp.228-33.
Wan, H., Song, X., Chen, G. & Gu, M., 2010. A Refinement-Based Validation Method for Programmable Logic Controllers. In *10th International Conference on Quality Software*. Zhangjiajie, 2010. pp.361-64.
Zurawski, R. & Zhou , M., 1994. Petri nets and industrial applications: a tutorial. *IEEE Transactions on Industrial Electronics*, 41(6), pp.567–83.

RESPONSABILIDADE AUTORAL

Os autores são os únicos responsáveis pelo conteúdo deste trabalho.

FRAMEWORK FOR THE DEVELOPMENT OF SIS CONTROL PROGRAMS BASED ON THE IEC61511 STANDARD

Rodrigo Cesar Ferrarezi, rferrarezi@usp.br[1]
Reinaldo Squillante Júnior, reinaldo.squillante@usp.br[1]
Jeferson A. L. Souza, jeferson.souza@usp.br[1]
José Reinaldo Silva, reinaldo@usp.br[1]
Fabrício Junqueira, fabri@usp.br[1]
Paulo Eigi Miyagi, pemiyagi@usp.br[1]
Lucas Antonio Moscato, lamoscat@usp.br[1]
Diolino J. Santos Filho, diolinos@usp.br[1]

[1] Polytechnic School – Department of Mechatronics Engineering and Mechanical Systems
University of São Paulo ... Av. Prof. Mello Moraes, 2231 - Cidade Universitária - São Paulo-SP - Brasil

Abstract*. Due to the high complexity of the actual Productive Systems, the design of suitable control systems according to the applicable industrial standards, and the possible negative impacts on human beings, on the environment and on equipment, the development of control solutions are both - secure and stable - is very demanded. One way to develop safer and more reliable systems is the use of Safety Instrumented Systems (SIS) according to the standards IEC 61508 and IEC 61511. However, even when the control programs are developed according to the applicable standards SIS control programs are prone to specification and design errors, as on the developing of any kind of software. One way to improve the reliability of these control programs, method which is also required by the safety standards IEC 61508 and IEC 61511 is the application of the SIS safety programs development cycle, together with the use of formal verification techniques on the control software models as well the use a unified approach for modeling the SIS prevention and mitigation systems, and thus having a better understanding of their interactions.*

Keywords*: IEC 61511, prevention and mitigation SIS, Model Checking, GHENeSys*