JOÃO BATISTA CAMARGO JUNIOR

ESTUDO DA SEGURANÇA EM SISTEMAS DE CONTROLE METRO-FERROVIÁRIOS

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Doutor em Engenharia

São Paulo
1996

JOÃO BATISTA CAMARGO JUNIOR

ESTUDO DA SEGURANÇA EM SISTEMAS DE CONTROLE METRO-FERROVIÁRIOS

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do título de Doutor em Engenharia

Área de Concentração:
Engenharia de Computação e Sistemas Digitais

Orientador
Prof. Dr. Benício José de Souza

1996

Aos meus pais

AGRADECIMENTOS

Ao amigo e orientador Prof. Dr. Benício José de Souza, pela orientação segura e precisa, possibilitando a elaboração deste trabalho.

Aos colegas da FDTE e da Escola Politécnica, que de alguma forma tenham contribuído para a realização deste trabalho.

Ao meu amigo Jorge Rady de Almeida Júnior, pelo seu companheirismo.

Aos meus amigos, que sempre me incentivaram na realização deste trabalho.

A todos os meus familiares, pela compreensão, paciência e incentivo à elaboração desse trabalho.

SUMÁRIO

RESUMO

Este trabalho apresenta uma contribuição para a avaliação da segurança de sistemas críticos, especialmente os de controle metro-ferroviários, através de uma nova metodologia com enfoque primordial para a questão da completeza das suas especificações.

Esta metodologia tem como base de aplicação o modelo de transição de estados, sendo seus critérios de verificação denotados através de expressões WFF acrescidas de funções típicas ao modelo utilizado.

Este método de avaliação é aplicado a um controle metro-ferroviário típico, obtendo-se resultados promissores como ferramenta de análise de segurança de sistemas críticos.

ABSTRACT

This works presents a contribuition to the evaluation of safety critical systems, especially the metro-railways controls, through a new methodologie with primordial focus in the completeness of their specifications.

This methodologie uses a state transition model to specify an application, with its verifications criteria denoted through WFF expressions added by typical functions of the used model.

This method of evaluation is applied to a typical metro-railway control, showing promising results as an safety analysis tool for critical systems.

1. INTRODUÇÃO

Este trabalho trata da segurança de sistemas de controle metro-ferroviários que utilizam técnicas computacionais. A preocupação com a segurança de sistemas não é totalmente nova. Em 1800, James Watt foi contra o uso de bombas de alta pressão devido ao perigo de explosão. Thomas Edison, analogamente, desaconselhou o uso de eletricidade de alta voltagem em 1880. Entretanto, as conseqüências de acidentes no passado eram limitadas, situação bem diferente hoje, em que um acidente pode ter conseqüências bastante graves.[STANKOVIC 88] Em função da experiência nessa área do conhecimento, pode-se dizer que a maioria dos acidentes não é devida à falta de conhecimento de princípios científicos, mas a deficiências na aplicação de técnicas de engenharia. Outro princípio que se tira dessa experiência é que a prevenção de acidentes não se dará apenas através da tecnologia computacional, mas também por intermédio de aspectos relacionados com a produção e a operação dos sistemas.[LEVESON 95]

Define-se, de um modo genérico, um sistema crítico como sendo aquele que exige um projeto adequado para se obter uma operação segura. Dentre os sistemas críticos, pode-se citar os controladores de vôo, sistemas militares, alguns controles industriais, usinas nucleares, indústrias químicas, sistemas de controle metro-ferroviários, entre outros.

Um importante fator a se destacar é que a segurança, nos sistemas críticos modernos, não está apenas ligada ao software de controle, mas ao sistema como um todo. Isso implica dizer que, mesmo com a existência de um "software correto", não se obtém, necessariamente, um sistema seguro. Esses sistemas devem ainda ser tolerantes a defeito, principalmente quando esse defeito resulta numa falha insegura.

Este trabalho tem como objetivo principal apresentar uma metodologia de avaliação da segurança de sistemas críticos, aprofundando-se na fase de especificação e tendo como enfoque primordial a questão da completeza das especificações, considerando o ambiente da aplicação e o próprio sistema de controle. Essa metodologia tem como suporte o modelo de transição de estados, sobre o qual são aplicados critérios para verificação da completeza dessa especificação. A preocupação com esse atributo da especificação de sistemas críticos reside no fato de a maioria dos acidentes ter, como causa fundamental, a omissão nas especificações desses sistemas de condições que podem levá-lo a situações inseguras. Complementando este objetivo, e não menos importante, existe a aplicação desta metodologia em sistemas de controle metro-ferroviários, visando destacar aspectos fundamentais no desenvolvimento seguro de tais sistemas. Vale ressaltar também que a

cobertura dos critérios propostos é otimizada não só através de uma análise objetiva da formalização do modelo de transição de estados, mas também por intermédio de uma avaliação dos resultados obtidos num sistema de controle que envolve aspectos de segurança. Dessa forma, esta metodologia procura também sistematizar e estruturar a avaliação da segurança de sistemas críticos, tornando-a mais objetiva, criteriosa e completa.

O trabalho é apresentado em 6 (seis) capítulos.

O capítulo 2, "Histórico e Motivação", apresenta um breve histórico de como se originou o conceito de segurança, apresentando também diversos trabalhos nessa área, tanto nos Estados Unidos como na Europa. Finalizando esse capítulo, é destacada a preocupação atual com a segurança nos sistemas de controle metro-ferroviários no Brasil.

O capítulo 3, "Conceitos Básicos de Sistemas Tolerantes a Defeito", apresenta as definições principais na área de segurança e confiabilidade, fundamentais para o perfeito entendimento da metodologia proposta no capítulo seguinte. Além desses conceitos, é apresentado também um Programa Global de Avaliação da Segurança de sistemas críticos, dentro do qual deve-se enquadrar a metodologia sugerida no capítulo 4 deste trabalho.

No capítulo 4, "Proposta de Metodologia de Avaliação da Segurança de Sistemas Críticos", são apresentadas as necessidades e os fundamentos que levaram à criação do método de avaliação de segurança proposto, baseado no modelo de transição de estados e tendo, como enfoque principal, a completeza da especificação dos sistemas críticos. São mencionadas também, neste capítulo, as justificativas que levaram à adoção do modelo de transição de estados como plataforma de desenvolvimento deste trabalho.

O capítulo 5, "Avaliação da Metodologia Proposta", apresenta, inicialmente, a evolução tecnológica por que passam os sistemas de controle metro-ferroviários no mundo, destacando a necessidade crescente de rígidos critérios de segurança, em função da crescente complexidade desses sistemas. Em seguida, a metodologia proposta no capítulo 4 é aplicada num sistema modelo de controle metro-ferroviário, visando destacar certos aspectos fundamentais na obtenção da segurança desses sistemas de controle.

A "Conclusão" deste trabalho é apresentada no capítulo 6, destacando as contribuições, os resultados práticos obtidos e apontando os aspectos que merecem uma continuidade de pesquisa e aplicação.

2. HISTÓRICO E MOTIVAÇÃO

Os primeiros trabalhos na área de segurança, como uma disciplina separada, iniciaram-se após a II Guerra Mundial, no campo da aviação. Nessa época, os sistemas de aviação não apresentavam boas condições de segurança, visto que entre 1952 e 1966 foram registrados, nos Estados Unidos, diversos acidentes, com perda de 7.715 aeronaves e 8.547 mortes.[LEVESON 95]

Em função da preocupação crescente com a segurança, começaram a surgir os primeiros congressos e seminários nessa área do conhecimento. Uma das primeiras ocasiões em que se usou o termo "System Safety" foi num seminário sobre aviação realizado em 1954 nos Estados Unidos.

Outro fator que alavancou a pesquisa sobre segurança de sistemas foi o desenvolvimento de mísseis balísticos intercontinentais, controlados e supervisionados automaticamente. Além desses projetos, que envolviam conceitos de segurança, o Departamento de Defesa Americano (**DoD**) e a Comissão de Energia Atômica também passaram a se preocupar com as questões relacionadas à segurança, em função do desenvolvimento de instalações nucleares. O grande esforço inicial nessa atividade foi o estabelecimento de critérios rígidos de segurança, devido às graves conseqüências de um possível acidente nesse tipo de aplicação.

A primeira especificação de sistemas de segurança foi criada pela Força Aérea Americana em 1966: a recomendação MIL-S-38130A. Em junho de 1969, esse documento tornou-se a norma "System Safety Program for Systems and Associated Subsystems and Equipament: Requirements for - MIL-STD-882" e foi estabelecido, pelo **DoD**, um Programa de Segurança a ser exigido em todos os seus produtos e sistemas.

Este Programa de Segurança estabelecia, de maneira global, os seguintes objetivos:

- os critérios de segurança devem ser consistentes com os requisitos da missão;
- devem ser identificados os perigos associados com o sistema, subsistema e equipamentos, além de avaliar e controlar os níveis de segurança aceitáveis;
- devem ser estabelecidos requisitos de controle sobre perigos que não podem ser eliminados, visando proteger as pessoas, os equipamentos e as propriedades;

- deve ser estabelecido um nível de máximo risco aceitável para a utilização de novos materiais e novos produtos;
- a inclusão de fatores de segurança durante a aquisição de um sistema deve minimizar as correções durante a operação desse sistema; e
- os dados históricos sobre segurança, gerados em sistemas similares, podem ser considerados em novos sistemas, quando apropriados.

À medida que os computadores passaram também a ser utilizados nessas áreas de segurança, devido ao aumento das necessidades exigidas por esses sistemas, o conceito de segurança passou a se relacionar adicionalmente com módulos de software. Esse novo aspecto de segurança despertou grande preocupação, tanto no **DoD** como na "National Aeronautics and Space Administration" (**NASA**).

Visando preencher essa lacuna, a **NASA** passou a investir em recomendações de segurança aplicadas especificamente sobre módulos de software. Dentro desse contexto, surgiram algumas normas.

A norma "Software Quality Assurance Audits Guidebook" descreve as atividades de auditoria para a obtenção da qualidade do software. Basicamente, essa norma abrange as seguintes atividades de auditoria:

- Planejamento e Preparação da Auditoria

  Nesta fase, deve ser elaborado um plano para examinar todos os processos de engenharia, gerenciamento e certificação do software de todos os produtos. Os processos de engenharia incluem análise, projeto e codificação. Os processos de gerenciamento incluem registro dos estados e gerenciamento da configuração, e os processos de certificação englobam a Verificação & Validação, além dos relatórios de não conformidade e ações corretivas.

- Visita Local

  Nesta etapa, são coletados os dados necessários para assegurar que o produto requerido está sendo produzido em concordância com as normas aplicáveis e que o estado apresentado nos relatórios corresponde ao estado real do produto.

- Registro da Auditoria

Nesta atividade, é apresentado um quadro bastante claro do estado das tarefas de desenvolvimento e gerenciamento do projeto. Em alguns casos, podem ser identificados certos procedimentos pré-estabelecidos que não estão sendo eficazes na melhoria da qualidade do produto.

- Acompanhamento

  Após o registro da auditoria, ações são essenciais para resolver deficiências identificadas anteriormente. As prováveis modificações devem ser rastreadas, para assegurar sua efetivação e documentação.

De forma complementar, a norma "Software Assurance Standard" [NASA-STD-2201-93], aprovada em novembro de 1992, especifica, de maneira global, um padrão para o Programa de Certificação de Software da **NASA**.

Esse programa, de forma resumida, determina as seguintes atividades:

- assegurar que os requisitos de certificação estão documentados e verificados ao longo do ciclo de vida do software;
- detectar condições existentes, ou em potencial, que podem degradar a qualidade do software, incluindo deficiências e incompatibilidades, além de apresentar ações corretivas a serem realizadas; e
- certificar da existência de ações preventivas eficazes e periódicas, no sentido de identificar as causas das deficiências e não conformidades.

Uma das atividades fundamentais desse Programa de Certificação denomina-se "Certificação da Segurança do Software", que implica verificar o atendimento dos Requisitos Gerais de Segurança alocados ao software, além da identificação e verificação dos controles de segurança que são implementados por software. Um dos subitens dessa Certificação de Segurança do Software explicita a necessidade de ser analisada a consistência, completeza, correção e facilidade de teste dos requisitos de segurança.

Essa norma, NASA-STD-2201-93, cita como referência outras normas, como por exemplo:

- "Software Formal Inspection Guidebook" - [NASA-GB-A302], aprovada em agosto de 1993 e que fornece informações de como implementar um método de inspeção

formal. Essas inspeções se realizam ao longo do processo de desenvolvimento do produto software; e

- "Software Formal Inspection Standard" - [NASA-STD-2202-93], aprovada em abril de 1993. Essa norma define os requisitos que devem ser atendidos num processo de inspeção formal do software.

Em função das necessidades crescentes da sociedade, outras áreas de aplicação, não militares, também começaram a desenvolver sistemas computacionais de controle que envolviam aspectos de segurança. Dentre essas novas aplicações, pode-se citar as Indústrias Químicas, Energéticas, Siderúrgicas, de transportes metro-ferroviários, de automação hospitalar, entre outras. Essas novas aplicações passaram a desenvolver seus próprios Programas de Segurança ou a adaptar o Programa de Segurança da **NASA**, já existente. Dessa maneira, a preocupação maior passou a ser a prevenção de acidentes ao invés do estudo das causas e de sua eliminação após a ocorrência de um determinado acidente.

Outro fator que colaborou e continua colaborando para a uma crescente automação de sistemas de segurança em outras áreas não militares, especialmente na área de transporte metro-ferroviário, diz respeito ao fim da guerra fria e à conseqüente utilização de todo o conhecimento tecnológico e científico já adquirido anteriormente. Todo esse desenvolvimento está resultando em sistemas de controle metro-ferroviários cada vez mais complexos, podendo-se citar o **ATCS** - "Advanced Train Control System", desenvolvido por pesquisadores americanos e canadenses oriundos da área militar. Na Europa, está sendo desenvolvido o **ETCS** - "European Train Control System", visando uma melhor integração entre os diversos sistemas existentes. Uma breve descrição de cada um desses sistemas é apresentada no capítulo 5 desta tese, dentro da evolução tecnológica dos sistemas de controle metro-ferroviários.

Evidentemente, em paralelo com essa evolução e com a crescente preocupação com os aspectos relacionados com a segurança, as universidades também passaram a investir em importantes pesquisas nessa área do conhecimento. Dentre esses trabalhos, pode-se destacar quatro linhas básicas de pesquisa, a saber:

- pesquisa de Programas Globais de Segurança estabelecendo critérios objetivos de segurança;

- pesquisas envolvendo a segurança de sistemas de software baseados em modelos probabilísticos;
- pesquisas envolvendo a segurança de sistemas de software baseados em desenvolvimentos formais; e
- pesquisas envolvendo a combinação de metodologias formais e informais com o objetivo de se obter sistemas com maior grau de segurança.

Além dessas diversas pesquisas realizadas passaram a existir, dentro das Universidades, diversos cursos especializados na área de segurança, além da crescente publicação de livros básicos a respeito deste assunto.[KECECIOGLU 84]

Especificamente na área metro-ferroviária, podem ser apresentados alguns esforços sendo realizados em nível mundial, no sentido de se obter um maior grau de segurança.

A "Federal Railroad Administration" (**FRA**) foi criada nos Estados Unidos em 1967 com as seguintes missões básicas:

- aumentar a segurança nas ferrrovias intermunicipais para transporte de passageiros;
- aumentar a segurança nas ferrovias para transporte de cargas; e
- aumentar a disponibilidade das ferrovias americanas.

Basicamente, a **FRA** visa o avanço tecnológico das ferrovias, almejando segurança, rapidez, eficiência, diminuição da poluição e aumento da confiabilidade. Além disso, essa organização determina algumas padronizações na área de segurança.

Para exemplificar a importância da pesquisa na área de segurança e confiabilidade, foram comunicados à **FRA**, em 1988, 3.051 acidentes nas ferrovias americanas, de acordo com [ORTH 94]. Com todo o trabalho que foi desenvolvido em conjunto com as pesquisas realizadas na área de segurança em 1989, o número de relatos de acidentes caiu para 2791 em 1993. Apesar do aumento no número de linhas e viagens em todo o país, o número de acidentes se manteve constante nos últimos anos e até caiu em 1993, mostrando a necessidade de se adotarem medidas rigorosas de segurança e metodologias de análise, verificação e validação desses sistemas.

Desses 2.791 acidentes relatados, tem-se como causas apontadas:

- 36% por defeitos na sinalização e rastreamento;
- 31% por fatores humanos;
- 13% por falhas no equipamento mecânico e elétrico;
- 6% por colisões com veículos automotivos em cruzamentos com rodovias; e
- 13% por outros fatores.

Como conseqüência, o custo total para as ferrovias com esses acidentes ficou da ordem de US$ 190 milhões.

Além desses aspectos, a **FRA** estabeleceu que quaisquer mudanças no equipamento ferroviário - velocidades de operação, procedimentos operacionais, entre outras - requerem uma nova avaliação, de forma a garantir que essas mudanças não irão comprometer a segurança nos transportes de carga e/ou passageiros, cujo sistema anterior já havia sido analisado e validado.

Estava prevista para 1995 uma adaptação das normas e padronizações da **FRA**, visando refletir a realidade das inovações tecnológicas nas áreas de sinalização, rastreamento, propulsão, controle microprocessado, entre outras.

Uma outra organização americana, a "Association of American Railroad" (**AAR**), foi criada em 1934 com o intuito de prover centralização efetiva de autoridade sobre assuntos de interesse comum, protegendo e inovando a indústria ferroviária.

O conceito de atuação cooperativa é muito utilizado nessa associação. Essa cooperação se dá através da elaboração de objetivos comuns entre diversas entidades. Essas entidades, além das próprias ferrovias associadas, muitas vezes são caracterizadas por indústrias fornecedoras de componentes e sistemas, universidades e empresas de consultoria particulares. Um exemplo desse tipo de cooperação se dá através de contratos entre a **AAR** e universidades, envolvendo pesquisas, testes e desenvolvimento, muitas vezes relacionado à construção e aplicação de metodologias de testes e validação de hardware e software aplicativo. Além dos contratos comuns, muitas vezes essas associações envolvem a construção de laboratórios.

Algumas das universidades que participam ativamente dessa cooperação, segundo [LUNDGREN 94], são: University od Massachusetts, Iowa State University, University of Michigan, University of New Hampshire, Oregon Graduate Institute, Georgetown University e Illinois Institute of Technology.

Na Europa, da mesma forma, existem grupos trabalhando na área de padronização metro-ferroviária visando uma maior segurança [GLORIA 94]. Os grupos **CEN** e **CENELEC**, ligados à "Community of European Railways - **CER**", foram criados em 1989. O **CEN** é dedicado a aplicações ferroviárias não-elétricas e o **CENELEC**, para aplicações elétricas e eletrônicas em geral. Os comitês técnicos mais importantes que estão operando são o **TC256** do **CEN** e o **TC9X** do **CENELEC**.

A proposta básica desses comitês é a de fornecer suporte à abertura do mercado europeu através do estudo das padronizações já existentes em cada nação participante, tentando-se chegar a padronizações em nível internacional, criando produtos europeus ao invés de produtos nacionais. Alguns objetivos são: redução de custos por vasta aplicabilidade de componentes, alta competitividade entre fornecedores, procedimentos comuns de qualificação e testes, interoperabilidade entre produtos e sistemas, etc.

Quanto à padronização de parâmetros genéricos de segurança e confiabilidade, existem problemas devido às novas tecnologias. Enquanto os índices de confiabilidade das tecnologias mais antigas eram mais fáceis de serem padronizados e, na prática, determinados através de métodos consolidados de testes e validação, as tecnologias novas envolvem diversos problemas, em que o enfoque tradicional de cálculo de índices de segurança e confiabilidade não são facilmente aplicáveis. Alguns desses problemas relacionam-se com a complexidade crescente dos componentes utilizados.

Para o futuro, fala-se na padronização de softwares para o controle de ferrovia. Com isso, pretende-se determinar critérios de segurança para sistemas de sinalização e produtos e permitir a especificação, a validação e o acesso a protótipos.

Dentro do comitê técnico **TC9X** do **CENELEC**, existe um grupo denominado **SC9XA**, encarregado da criação de normas em termos de sistemas de comunicação, de sinalização e processamento. Dentro desse grupo, existe um subgrupo denominado **WGA1**, encarregado da normalização do software. O **WGA1** gerou em fevereiro de 1994 a primeira versão da norma européia "Railway Applications: Software for Railway Control and Protection Systems - EN 50128". Esse documento normaliza técnicas, medidas e

documentos a serem gerados durante todo o desenvolvimento de um software para aplicações ferroviárias.

Alguns dos propósitos básicos desta norma são:

- minimizar os riscos de projeto, eliminando situações perigosas ou reduzi-las a um nível aceitável;
- incorporar ao software alguns dispositivos de segurança, de forma a reduzir riscos residuais; e
- prever indicações da "atenção" caso algum risco residual ainda estiver presente em um nível aceitável.

Um problema encontrado foi justamente o fato de a utilização de novas tecnologias tornar difícil a obtenção de resultados quantitativos em termos de análise e validação. Dessa forma, essas novas tecnologias requerem um processo mais formal de verificação e validação da segurança.

Essa norma se baseia também em níveis de integridade do software, que são medidos em função dos níveis de risco aceitáveis. A especificação desses níveis deve seguir o seguinte critério:

- identificação dos níveis de integridade do sistema através da classificação dos riscos e identificação dos critérios de verificação associados aos níveis de risco aceitável; e
- divisão dos níveis de integridade do sistema entre os subsistemas, definindo os níveis relacionados.

É apresentado também, nessa norma, um destaque muito grande quanto a importância da independência entre os vários papéis desempenhados ao longo do ciclo de vida do software, ou seja: projetista, testador, verificador, validador e analista. Além desse aspecto, é recomendada a verificação e validação do software em todos os níveis de integridade.

Toda essa preocupação com a padronização visa permitir uma adequada integração dos projetos dentro da comunidade européia dentro de níveis aceitáveis de segurança e confiabilidade.

Evidentemente, todos esses trabalhos internacionais influenciaram e continuam influenciando as pesquisas e os projetos de controle metro-ferroviário no Brasil. Essa preocupação com a segurança e a confiabilidade dos sistemas de controle metro-ferroviários fica bastante evidenciada, por exemplo, através da "Carta de Conclusões do Congresso Internacional de Sinalização, Telecomunicações e Energia - Câmara de Sinalização", organizado pelo Comitê Brasileiro Metro-Ferroviário da ABNT, realizado em agosto de 1993 no Rio de Janeiro. Algumas das recomendações contidas nessa carta são abaixo citadas:

- "que os novos sistemas a serem implantados sejam dotados em sua concepção de recursos técnicos que permitam a previsão de diagnósticos precoces de falha;

- que as empresas operadoras avaliem a viabilidade de implantação de sistemas automáticos de detecção, identificação e rastreamento de composições ferroviárias via rádio com recursos de blocos fixos ou móveis com a devida comprovação de confiabilidade, disponibilidade e segurança;

- que as empresas operadoras definam formalmente em suas especificações técnicas os Requisitos Gerais de Segurança (**RGS**) do sistema a ser adquirido;

- considerando-se que, atualmente, a análise de segurança de sistemas está sendo concebida segundo os seguintes critérios:
  - alguns fabricantes e operadoras estrangeiras fazem a sua própria análise;
  - alguns fabricantes e operadoras contratam análise independente;
  - algumas operadoras consideram suficiente a análise do fornecedor;

  e que não há ainda uma padronização desta questão em nível mundial, recomenda-se que o Comitê Brasileiro Metrô-Ferroviário-**CB06** promova um estudo específico sobre o assunto envolvendo operadoras, fabricantes e empresas de consultoria do setor, a fim de se obter uma normalização do Modelo de Análise de Segurança junto à ABNT;

- que seja adotada como premissa básica pelas empresas operadoras a necessidade de efetuar-se uma análise de segurança para todo produto que

desempenhe funções de segurança, concebido diferentemente do similar consagrado;

- que a documentação de sistemas que desempenhem funções de segurança seja padronizada, tanto ao nível do hardware quanto ao nível do software, conforme recomendações a serem determinadas pela ABNT;

- que todo enlace de comunicação via rádio, pelo qual trafeguem informações de segurança, apresente características vitais com relação ao aspecto de tolerância a falhas e entradas impróprias."

Ainda mais recentemente, no Congresso Internacional de Sistemas de Controle e Integração Tarifária organizado pelo Comitê Brasileiro Metro-Ferroviário da ABNT, realizado em outubro de 1995, em São Paulo, podem ser destacadas as seguintes recomendações:

- "que as empresas operadoras definam claramente em suas especificações técnicas os Requisitos Gerais de Segurança (**RGS**) e os Requisitos Gerais de Operação (**RGO**) do sistema; e

- que as operadoras de sistemas metro-ferroviários exijam formalmente, em suas especificações, a análise de segurança do sistema. A forma de contratação e avaliação deverá ser objeto de acordo entre o comprador e o fornecedor, uma vez que não existe atualmente uma padronização dessa questão em nível mundial."

Em função de todas essas considerações, da experiência adquirida na área de controle metro-ferroviário e da grande necessidade de pesquisas sobre segurança e confiabilidade, existe uma enorme motivação na realização desta tese de doutorado. Evidentemente, além desses aspectos, há uma grande satisfação em poder colaborar, de maneira objetiva, na melhoria da qualidade dos serviços de transporte em nosso País.

3. CONCEITOS BÁSICOS DE SISTEMAS TOLERANTES A DEFEITO

Neste capítulo são apresentados alguns conceitos básicos sobre Sistemas Tolerantes a Defeito, destacando os principais aspectos dessa área de conhecimento e que são importantes para a compreensão da metodologia proposta no capítulo 4, bem como da sua aplicação apresentada no capítulo 5 desta tese.

3.1 Definições Fundamentais

Neste item, são apresentados alguns conceitos fundamentais, destacando a Confiabilidade, a Disponibilidade, a Segurança, o Desempenho, a Manutenabilidade e a "Testabilidade".

Outro aspecto importante neste trabalho é a padronização de termos técnicos em português, como por exemplo os conceitos de "Defeitos", "Erros" e "Falhas", correspondentes aos termos "Fault", "Error" e "Failure".

A confiabilidade **R(t)** de um sistema corresponde à probabilidade condicional de ele desempenhar corretamente no intervalo de tempo **[t0,t]**, dado que o sistema estava funcionando corretamente no instante **t0**. Trata-se de uma probabilidade condicional, pois ela depende de o sistema estar operacional no início de um intervalo de tempo escolhido.

A Disponibilidade **A(t)** é a probabilidade de o sistema estar operando corretamente e estar disponível para realizar as suas funções no instante de tempo **t**. A Disponibilidade difere da confiabilidade no aspecto fundamental de que a confiabilidade depende de um intervalo de tempo, enquanto a disponibilidade é calculada em um instante de tempo.

Define-se como estado inseguro um estado em que o sistema está exposto a acidentes sem a necessidade da ocorrência de falhas ou entradas impróprias.

A Segurança **S(t)** é a probabilidade condicional de o sistema não atingir um estado inseguro no intervalo de tempo **[t0,t]**, dado que o sistema estava num estado seguro no instante **t0**. A segurança é uma medida da capacidade de o sistema ser "Fail-Safe", ou seja, quando falhar, falha de maneira a não comprometer a segurança do mesmo. No capítulo 4, o conceito de segurança será estendido para todos os estados de um sistema.

O Desempenho de um sistema corresponde à probabilidade de um subconjunto de funções estar operando corretamente num determinado instante de tempo. Esse subconjunto de funções está relacionado com o nível de desempenho do sistema.

A Manutenabilidade de um sistema corresponde à facilidade com a qual ele pode ser reparado, dado que ele falhou. Nesse sentido, é a probabilidade de o sistema, em falha, ser reparado dentro de um intervalo de tempo.

A "Testabilidade" do sistema corresponde à habilidade oferecida para se testarem certos atributos. Dessa forma, mede a facilidade com a qual certos testes podem ser realizados.

São apresentadas, a seguir, algumas definições importantes para uma melhor compreensão dos mecanismos tolerantes a **defeito**. Um dos grandes problemas dos sistemas de controle é como construir mecanismos que evitem que ele **falhe**. Além disso, outro problema é como localizar **erros** internos ao sistema.

Apenas analisando este último parágrafo, sente-se a necessidade de definir termos que são muito utilizados nessa área de conhecimento, tais como **defeito, erro** e **falha**. As definições que se seguem [JOHNSON 89] não são padronizadas, existindo outras recomendações [PFLEEGER 92].

O **defeito** corresponde a uma imperfeição que ocorre dentro de um componente, seja ele hardware ou software. Pode-se ilustrar esse fato com um condutor aberto, uma imperfeição física num dispositivo semicondutor, ou um "loop" infinito num programa de software. O defeito está relacionado, dessa forma, com o universo físico.

O **erro** corresponde à manifestação de um defeito, contaminando a informação interna do sistema, desviando-a de sua correção ou de sua precisão. O erro está relacionado com o universo da informação.

A **falha**, por sua vez, representa uma exteriorização do erro, ou seja, o sistema passa a não desempenhar suas funções corretamente. A falha está relacionada com o universo externo, ou seja, com o ambiente da aplicação.

A figura 3.1 apresenta a seqüência lógica entre esses termos.

Figura 3.1 - Relação entre defeito, erro e falha

A duração de tempo entre a ocorrência de um defeito e o aparecimento do erro correspondente denomina-se Latência do Defeito.

A duração de tempo entre a ocorrência de um erro e o aparecimento de uma falha denomina-se Latência do Erro.

Esses conceitos de tempo de latência são fundamentais, já que a falha pode ser evitada através de mecanismos utilizados durante esses intervalos de tempo [SHIN 84].

3.1.1 Caracterização dos Defeitos

A caracterização dos defeitos deve ser bastante objetiva, facilitando, desse modo, a determinação de cuidados especiais para evitá-los, bem como a determinação de técnicas para tolerá-los [LAPRIE 92].

Os defeitos podem ser caracterizados de acordo com 5 (cinco) características básicas:

- Causa;
- Natureza;
- Duração;
- Extensão; e
- Valor.

Os defeitos podem resultar de uma variedade de fatores que ocorrem dentro de componentes, externos aos mesmos ou durante o projeto de componentes ou de todo um sistema.

As principais grandes áreas relacionadas com a causa dos defeitos são:

- problemas na Especificação;
- problemas na Produção;
- desgaste do Componentes; e
- fatores Externos.

Os defeitos no hardware podem ter sua origem em qualquer uma das áreas apontadas anteriormente, enquanto que os defeitos no software teriam sua origem apenas em problemas na especificação e na implementação. Como o componente software não envelhece, ele não apresenta problemas devido ao desgaste, além de não sofrer interferências devido a fatores externos, como interferência eletromagnética, ruídos aleatórios, entre outros. Nesses casos, o software pode não funcionar corretamente, mas tem como provável causa algum problema original no componente de hardware que o executa.

A Natureza dos defeitos pode ser classificada em:

- Defeitos de Hardware
    - Analógico
    - Digital
- Defeitos de Software

A Duração dos defeitos pode ser organizada da seguinte forma:

- Permanente
- Intermitente
- Transitório

A Extensão do defeito pode ser:

- Local
- Global

O valor do defeito pode ser classificado como:

- Determinado
- Indeterminado

Todo esse detalhamento do defeito auxilia na determinação das técnicas adequadas para evitá-lo e, em última instância, tolerá-lo.

Um dos grandes problemas em sistemas de segurança são as falhas de modo comum. Este tipo de falha ocorre quando múltiplas cópias de um sistema redundante sofrem defeitos quase que simultaneamente, geralmente devido a uma única causa, podendo ser defeitos de hardware ou software. De acordo com [HETCHT 87], de 17 a 18% das anomalias em sistemas computacionais redundantes são produzidas por falhas de modo comum.

3.1.2 Filosofias de Projeto para Combater os Defeitos e os Erros

Como já se havia apresentado no item anterior, as falhas são exteriorizações dos erros, que por sua vez são conseqüências dos defeitos. Baseadas nesses conceitos básicos, existem três filosofias de projeto que podem ser utilizadas visando combater os defeitos e os erros:

- evitar Defeitos;
- mascarar Defeitos; e
- tolerar Defeitos.

De forma bastante suscinta, a filosofia de evitar defeitos engloba atividades como revisões de projeto, testes, métodos de controle de qualidade, métodos preventivos, diversidade de projeto, métodos formais, entre outros [LALA 94].

A filosofia de mascarar defeitos não se preocupa com a detecção do defeito, mas sim com o seu mascaramento, através de técnicas como a Correção do Erro e Votação Majoritária.

Por outro lado, a filosofia de Tolerar Defeitos utiliza-se, além do mascaramento de defeitos, também do conceito básico de Reconfiguração, ou seja, a detecção e

localização do defeito/erro com a posterior remoção do componente defeituoso. Por trás desse conceito, está a filosofia de Redundância. Basicamente, a Reconfiguração compõe-se das seguintes etapas:

- detecção do defeito/erro;
- localização do defeito/erro, através da qual será direcionada a recuperação;
- isolamento do defeito/erro, para previnir que seus efeitos propaguem através do sistema; e
- recuperação do defeito/erro através da substituição do componente causador do defeito/erro.

3.2 Aplicações na Vida Real

São apresentados neste item as principais áreas de aplicação dos Sistemas Tolerantes a Defeito, destacando as Aplicações de Vida Longa, as Aplicações Críticas, as Aplicações com Manutenção Adiantada/Adiada e Aplicações de Alta Disponibilidade.

As aplicações de Vida Longa apresentam requisitos de disponibilidade em torno de 0,95 depois de 10 anos de operação. No entanto, podem suportar pequenos períodos fora de funcionamento. Nesse tipo de sistema, enquadram-se os sistemas a bordo de naves e satélites [SCHNEIDEWIND 92].

As Aplicações Críticas relacionam-se com o conceito de segurança, destacando-se os Controladores de Vôo [PERRY91], [CRAIGEN2 94], Sistemas Militares, alguns Controles Industriais, Usinas Nucleares [HUSSEINY 90], [CRAIGEN1 94], Indústria Química, Aplicações Hospitalares [MOJDEHBAKHSH 94], Sistemas de Controle Metro-Ferroviários [GERHART2 94], entre outros. Normalmente, para esses sistemas, são exigidos valores bastante altos de MTBUF: em torno de dezenas e centenas de milhares de anos. O conceito de MTBUF é esclarecido no item 3.4.

As aplicações com Manutenção Adiantada ou Adiada correspondem aos sistemas em que as operações de manutenção são extremamente custosas, inconvenientes ou difíceis de se realizarem. Nesse tipo de sistema, encaixam-se as Estações de Processamento Remoto (como por exemplo, estações de transmissão distantes), Satélites, entre outros. Nessas aplicações, são realizadas visitas periódicas com o objetivo de efetuar manutenções

preventivas. Durante os períodos entre as visitas, o sistema manipula, de maneira automática, os problemas ocorridos.

Nas aplicações de Alta Disponibilidade, o usuário final deseja ter o sistema disponível, com uma alta probabilidade, quando solicitado. Um exemplo bastante adequado desse tipo de aplicação são os Sistemas Bancários.

3.3 Técnicas de Projeto para Alcançar Tolerância a Defeito

O conceito fundamental para se alcançar o requisito de "tolerância a defeito" está baseado na técnica de Redundância, ou seja, a simples adição de informação, além da que é necessária, para a operação normal do sistema.

A utilização da redundância pode fornecer uma capacidade adicional ao sistema, mas pode também ter um impacto importante no seu tamanho, peso, consumo e custo. Todos esses fatores devem ser considerados na decisão de projeto [AVIZIENIS 78], [ZAHEDI 91].

3.3.1 Redundância de Hardware

São apresentadas nesse item algumas técnicas de redundância de hardware. Evidentemente, outras arquiteturas podem existir, principalmente através de combinações de características peculiares a cada técnica aqui apresentada. [JOHNSON 89], [SIEWIOREK 74]

3.3.1.1 Votação Majoritária

Esse tipo de redundância utiliza-se do conceito de mascaramento de defeitos, para esconder a ocorrência do mesmo, não requerendo nehuma ação do sistema ou do operador. A redundância de hardware desse tipo mais comum denomina-se Redundância Modular Tripla (**TMR**), podendo ser aplicada também a software. Nessa arquitetura, três módulos processam as informações a serem avaliadas por um votador. A maior dificuldade no sistema **TMR** recai sobre o votador, ou seja, a segurança desse sistema não consegue ser melhor do que a segurança do votador.

Existem outras arquiteturas dentro desse tipo de redundância que simplesmente triplicam o estágio do votador, fornecendo três resultados independentes. Nesse caso, deve haver

uma votação final realizada por algum atuador do sistema de controle. Outras arquiteturas são denominadas multiplexadas adaptativas, em que é diminuída a taxa de utilização de cada módulo através da técnica de multiplexação [MASSUR 91].

Uma generalização do sistema **TMR** é a Redundância N-Modular, onde **N** módulos são votados, sendo **N** um número ímpar, para não haver a possibilidade de empate na votação.

Existe também uma classificação com relação ao tipo de solução a ser adotada na votação majoritária: votação síncrona das entradas e combinação das entradas de maneira assíncrona [SHIN 89]. Na votação síncrona, têm-se uma implementação mais simples, mas se requer um "overhead" no processo de sincronização. Na combinação das entradas de maneira assíncrona, existe o problema da especificação e implementação de algoritmos adequados para esse tratamento. Esse último caso é bastante dependente da aplicação mas, por outro lado, é mais compatível com o não sincronismo real dos sistemas distribuídos.

Outro aspecto importante a ser considerado na votação majoritária está baseado na assertiva de que todos os módulos, que não estão em falha, irão produzir a mesma saída. Esse aspecto só deve ser verdadeiro quando esses módulos utilizam as mesmas entradas. Para contornar esse problema, pode haver a troca de mensagens entre os módulos votantes, visando, dessa forma, aumentar a sua segurança.[LAMPORT 82]

3.3.1.2 Duplicação com Comparação

Nesse tipo de redundância, dois módulos processam as informações em paralelo, sendo que as saídas são enviadas a um comparador. Na maioria dos casos, é detectada a existência do defeito, mas não é tolerado, pois não há meios de saber qual dos módulos está com defeito. O conhecimento do módulo com defeito irá depender fundamentalmente do fator de cobertura de defeito do módulo em questão, que corresponde à capacidade do mesmo em se autodiagnosticar, ou através de um detector de defeito externo ao módulo.

3.3.1.3 Unidade Sobressalente

Aqui podem existir **N** módulos sobressalentes. A reconfiguração pode ser vista, nesse caso, como uma chave cuja saída é selecionada de um dos módulos bons. Se alguns ou

todos os módulos estão bons, pode-se fixar um esquema de prioridades. Qualquer módulo com defeito é eliminado dessa consideração através do bloco detector de defeito. Nesse tipo de arquitetura, o sistema precisa tolerar uma pequena interrupção durante o processo de reconfiguração. Em função do tempo necessário para essa reconfiguração, o sistema pode ser classificado como "hot-standby" ou "cold-standby". Como exemplo de sistemas "hot-standby", pode-se citar o controle de uma reação química, enquanto que, para sistema do tipo "cold-standby", têm-se os sistemas de satélite. Nos sistemas 'hot-standby", as unidades sobressalentes são energizadas e estão continuamente atualizadas a respeito do estado do sistema.

Uma questão chave nesse tipo de redundância é a capacidade de detectar o defeito num determinado módulo. Esse tipo de detecção pode ser realizado através da utilização de comparadores na saída de dois módulos. Quando a saída discorda, ambos os módulos são descartados e utilizados outros dois módulos, a não ser que o fator de cobertura de defeito identifique o módulo em falha. Nesse caso, é eliminado somente esse módulo falho.

3.3.1.4 Temporizador "Watchdog"

Uma técnica bastante utilizada nesse tipo de redundância de hardware é o "watchdog", em que um temporizador deve ser reiniciado numa base de tempo repetitiva. O defeito em realizar essa ação resultará na reiniciação ou finalização do sistema, conforme for o mais adequado e seguro, antes que outro defeito aconteça no sistema.

3.3.2 Redundância de Informação

A Redundância de Informação corresponde à adição de informação, para permitir a detecção, o mascaramento e, possivelmente, a tolerância a defeito. Neste item, são apresentados alguns tipos de redundância de informação utilizados.[JOHNSON 89], [SIEWIOREK 74]

Basicamente, em sistemas computacionais, a redundância de informação tem por objetivo básico aumentar a distância de código entre palavras de código válidas. A distância de código é definida como a mínima distância de Hamming entre duas palavras de código. A distância de Hamming corresponde ao número de posições de bits em que as duas palavras diferem. Ora, para se aumentar essa distância de código, deve-se, conseqüentemente, aumentar a quantidade de informação no código válido, originando,

dessa forma, um novo código. As informações adicionadas podem estar misturadas ou não ao código original. Caso essas informações não estejam misturadas, o código é dito separável; caso contrário, é denominado não-separável. O processo de codificação e decodificação é mais complexo quando se trata de um código não separável.

A redundância de informação mais conhecida é o codigo de paridade, muito utilizado em mémorias, podendo ser por palavra, por byte, por componente ou por múltiplos componentes, horizontal ou vertical. Trata-se de um código separável.

Outro tipo de redundância de informação são os códigos duplicados, utilizados em sistemas de comunicação e memória. São códigos separáveis. Os códigos duplicados podem apresentar diversas maneiras de se implementarem, ou seja, armazenando a mesma informação em outro endereço, de forma complementar, invertendo as metades menos significativa e mais significativa da informação original, entre outras formas.

O "Checksum" trata-se de uma redundância de informação, também bastante utilizada em sistemas tolerantes a defeito e do tipo separável. É muito aplicável quando blocos de dados devem ser transmitidos de um ponto para outro. O conceito básico que fundamenta essa redundância é a soma dos dados originais, que pode ser feita de diversas maneiras, como ignorando ou considerando o "overflow" da soma, somando pares de dados, ou adicionando o "overflow" ao resultado final da soma. Essas diversas formas existem visando aumentar a capacidade de detecção do defeito.

Outro tipo de redundância de informação muito utilizada são os denominados **CRC's**, ou códigos cíclicos, caracterizados pelo seu polinômio gerador **G(x)** de grau maior ou igual a **(n-k)**, onde **n** é o número de bits contidos na palavra codificada produzida por **G(x)**, e **k** é o número de bits da informação original a ser codificada. Esse tipo de redundância, não separável, tem a capacidade de detectar todos os erros simples e múltiplos, adjacentes, que afetem menos do que **(n-k)** bits. A propriedade de detecção de erros dos códigos cíclicos é particularmente importante nas aplicações de comunicações, em que erros transitórios podem ocorrer.

Os Códigos Aritméticos são, por sua vez, utilizados nas verificações das operações aritméticas. Esses códigos podem ser utilizados na implementação de um processador "fail-safe", ou seja, através da verificação de cada operação aritmética da ULA determina-se a existência ou não de um erro no processamento. Para não haver problemas de sincronismo nesses processadores "fail-safe", a computação é continuada

em paralelo ao processo de verificação de informações anteriores. Evidentemente, outros recursos deverão ser utilizados para cobrir os defeitos nas operações lógicas.

O código de correção de erro de Hamming, código separável, é formado particionando os bits de informação em grupos de paridade e especificando um bit de paridade para cada grupo. A habilidade de localizar qual é o bit errado é obtida através da superposição dos grupos de bits.

3.3.3 Redundância por Tempo

A Redundância por Tempo não requer, em princípio, a adição de hardware extra na implementação. A grande aplicação desse tipo de redundância recai na detecção de Defeitos Transitórios [SOSNOWSKI 94] e Defeitos Permanentes.

O conceito básico da redundância temporal é a repetição do processamento de forma a permitir que um defeito seja detectado.[JOHNSON 89] Após um determinado número de repetições, os resultados são comparados, detectando-se a presença de eventuais erros. Sendo um erro detectado, o processamento pode ser realizado novamente para verificar se a discordância continua. Tal método de trabalho é interessante para a detecção de erros resultantes de defeitos transitórios, mas não protege contra erros decorrentes de defeitos permanentes.

Para ser possível a realização da detecção dos erros devido a defeitos permanentes, através de redundância por tempo, deve-se realizar dois processamentos das informações em instantes diferentes. Além disso, em um desses processamentos, a informação deve ser codificada através de uma função de codificação **c(x)** antes do processamento, sendo no final o resultado decodificado através da função inversa de **c(x).** Em seguida, os valores de ambos os processamentos são comparados. A idéia básica que fundamenta esse tipo de redundância considera que os defeitos permanentes provocam erros diferentes quando são codificados. Existem diversos tipos de funções de codificação como, por exemplo, função complemento, função deslocamento, função operandos trocados, entre outras.

3.3.4 Redundância por Software

Em aplicações computacionais, muitos meios de detecção e de tolerância a defeito podem ser implementados por software. A Redundância por Software pode ocorrer de

muitas formas, não precisando duplicar o programa por completo para se ter esse tipo de redundância. São apresentadas neste item algumas técnicas de redundância por software, destacando-se a Verificação de Consistência, a Verificação de Desempenho, N-Versões de Software e Blocos de Recuperação.[JOHNSON 89]

A Verificação de Consistência usa, o conhecimento *a priori*, sobre as características da informação a ser verificada na sua correção. Por exemplo, em algumas aplicações, é conhecido que uma determinada grandeza nunca deve exceder uma certa magnitude. Se o sinal exceder essa magnitude, isso indica a presença de algum erro. A verificação da consistência pode também ser implementada por hardware, mas é mais realizada por software. Para ilustrar esse tipo de redundância, pode-se citar a transferência de pacotes de dados, em que, no início de cada pacote, há um cabeçalho contendo o número de palavras dentro daquele pacote. Na recepção dessas informações, se houver um desacordo entre o número de palavras realmente recebidas e o número especificado no cabeçalho, é detectado um erro no sistema. Nesse caso, pode ser pedido uma retransmissão ou algum outro tratamento mais adequado.

A Verificação de Desempenho constata, por sua vez, se o sistema está dentro do comportamento esperado. Esse tipo de redundância é implementado através, por exemplo, de testes dos componentes do sistema, procurando verificar os seus respectivos desempenhos. Podemos citar aqui os testes de memória, testes da UCP, testes de comunicação, testes de entrada/saída em rotinas, entre muitos outros.

A redundância de software conhecida por N-Versões [HUDAK 93], [MEYER 93] constitui-se em projetar e codificar o software **N** vezes e comparar os **N** resultados produzidos por cada um desses módulos. Cada um destes é projetado e codificado por um grupo separado de programadores a partir de uma mesma especificação. Dentro dessa filosofia de trabalho, é esperado que **N** projetos independentes não irão gerar erros iguais. Dessa forma, quando ocorrer um erro, o mesmo não ocorre em todos os módulos, ou, se ocorrer, acontece de maneira diferente em cada uma das versões, fazendo com que os resultados finais sejam diferentes [LYU 93]. Existem algumas polêmicas em relação a essa técnica de redundância de software. A primeira é que os projetistas e codificadores tendem a cometer erros similares [KNIGHT 86], não se garantindo que versões independentes de um programa não terão defeitos comuns [TAI 93], [LAPRIE 90]. Outro aspecto a se considerar é que todas as **N** versões do software se originam de uma mesma especificação, sendo fundamental não apenas que ela esteja correta, mas também completa. Quando se trabalha com especificação formal, pode-se

utilizar dessa técnica de N-versões a partir de N especificações formais diferentes, mas com o mesmo conteúdo semântico. A diferença nessas especificações estaria no tipo de linguagem formal a ser utilizado [KELLY 91]. Existem diversos tipos de linguagem de especificação, podendo-se citar algumas: Z, VDM, SCR, Statecharts. [FRASER 91], [WILLIAMS 94]

A redundância de software conhecida por Blocos de Recuperação [HUDAK 93], [MEYER 93] constitui-se numa adaptação da técnica de redundância de hardware já apresentada no item 3.3.1.3 e denominada Unidade Sobressalente. Nesse tipo de redundância de software, apenas 1(um) módulo é utilizado em um determinado instante, sendo que as demais implementações por software são utilizadas apenas quando o sistema falha. Podem ser aplicadas ao controle do processo ou a uma base de dados [RANGANATH 93]. Nessa técnica de blocos de recuperação, o programa aplicativo é dividido em subprogramas denominados blocos. Cada bloco possui diversas implementações disponíveis que podem ser executadas. Os resultados de um bloco de recuperação são avaliados por um teste de aceitação, que verifica a correção destes. Havendo a aprovação desse bloco, é executado o próximo bloco do programa aplicativo. Se não houver a aprovação por parte do teste de aceitação, o estado anterior do processo a essa última execução é recuperado e um bloco alternativo é executado. Os blocos alternativos são executados até que os resultados sejam aprovados pelo teste de aceitação. Se nenhum bloco alternativo conseguir a aprovação pelo teste de aceitação, então o sistema falha por completo. De acordo com [LEE 93], esse tipo de técnica é bastante utilizado no sistema IBM/MVS, evitando que este falhe sob diversas condições de defeito no software. Existem modelos sendo estudados para avaliar a confiabilidade dos Blocos de Recuparação baseados nos eventos de erro observados durante a fase de teste [PUCCI 92].

Um trabalho comparativo entre as técnicas de N-versões e Blocos de Recuperação [KANOUN 93] concluiu que a primeira técnica é melhor, pois o impacto dos defeitos relacionados, ou seja, de modo comum, tem um maior reflexo na técnica de Blocos de Recuparação, visto que os testes de aceitação são bastantes específicos com a aplicação e menos generalistas.

Uma outra técnica de redundância de software refere-se à combinação do Bloco de Recuperação com a N-versões, denominada Bloco de Recuperação com Consenso [SCOTT 87]. Nessa técnica é necessário, além das diversas versões dos blocos e do teste de aceitação, um procedimento de votação. Se não há concordância na votação, então é

acionado um bloco de recuperação. Nesse caso, a saída do bloco com "melhor" versão é examinada pelo teste de aceitação. Se for aprovada, é considerada como a saída correta. Caso contrário, a próxima "melhor" versão é sujeita ao teste de aceitação. O conceito de "melhor" versão deve estar relacionado com a qualidade da saída do bloco.

Deve-se ressaltar aqui que, quanto maior for a formalidade no desenvolvimento do software, menor será a necessidade da utilização de sua redundância. Nesse sentido, aspectos problemáticos que ainda não foram adequadamente cobertos pelos formalismos existentes são discutidos no capítulo 4 deste trabalho.

3.4 Algumas Técnicas para Avaliar a Segurança nos Sistemas Tolerantes a Defeito

Nesse ítem, são apresentadas algumas técnicas para se avaliar a segurança de um sistema tolerante a defeito. Para uma melhor compreensão dos assuntos apresentados, torna-se importante a apresentação de alguns conceitos fundamentais.

- Taxa de Falhas e Função Confiabilidade

  A taxa de falhas é normalmente denotada por $\lambda(t)$, e corresponde ao número esperado de falhas por unidade de tempo. A função de confiabilidade, expressa por **R(t)**, pode ser calculada a partir da taxa de falhas $\lambda(t)$.

- Tempo Médio para Falhar - **MTTF**

  O **MTTF** corresponde ao tempo médio esperado para o sistema operar antes que a primeira falha ocorra. Formalmente, o MTTF pode ser calculado da seguinte forma:

$$\mathbf{MTTF} = \int_0^\infty \mathbf{R(t).\, dt}$$

- Tempo Médio para Reparo - **MTTR**

  Corresponde ao temo médio requerido para um sistema ser reparado.

- Tempo Médio Entre Falhas - **MTBF**

  O tempo médio entre falhas pode ser calculado através da seguinte expressão:

  MTBF = MTTF + MTTR

- Tempo Médio Entre Falhas Inseguras - **MTBUF**

  Analogamente, o tempo médio entre falhas inseguras pode ser calculado como:

  MTBUF = MTTUF + MTTR

  onde, MTTUF corresponde ao tempo médio para a primeira falha insegura do sistema. Como a grandeza dos valores de MTTUF (centenas ou milhares de anos) é bem maior do que a grandeza dos valores de MTTR (unidades ou dezenas de horas), pode-se dizer que:

  MTBUF ≅ MTTUF

  O cálculo do **MTBUF** pode ser realizado através da seguinte expressão:

  $$\mathbf{MTBUF} = \int_0^{\infty} \int \mathbf{S(t)} \,.\, \mathbf{dt} \,;$$

  onde **S(t)** representa o nível de segurança do sistema e pode ser determinado a partir do modelamento do sistema. A seguir, são apresentados dois modelos clássicos, Modelo Combinatório e Modelo de Markov, que podem ser utilizados para a determinação da função de segurança **S(t)** do sistema.

- Modelo Combinatório

  Esse modelo utiliza técnicas probabilísticas que enumeram os diferentes caminhos no qual o sistema pode permanecer operacional e seguro.[KAPUR 90] As probabilidades dos eventos que fazem com que o sistema permaneça operacional e seguro são calculadas para formar uma estimativa da segurança do sistema. A segurança de um sistema pode ser calculada em termos da segurança dos seus componentes individuais. Os dois modelos mais comuns na prática são o modelo série e o modelo paralelo. No modelo série, cada elemento do sistema é requerido operar de forma segura, para que o sistema seja considerado seguro. No modelo paralelo, apenas um dos elementos deve estar operando de forma segura. Na prática, os sistemas são típicas combinações de subsistemas série e paralelo. Esse tipo de modelo é de difícil aplicação em sistemas

críticos, em que a combinação série/paralelo é bastante complexa. Outra limitação desse modelo deve-se ao fato de não se poder considerar as atividades de manutenção na sua metodologia. Aplica-se apenas a sistemas que não passam por um processo de manutenção preventiva, preditiva ou corretiva.

- Modelo de Markov

  O modelo de Markov, diferentemente do modelo combinatório, permite modelar o processo de reparo que ocorre na maioria dos sistemas. Os dois conceitos principais no modelo de Markov são os estados e as transições. Os estados representam todas as situações que o sistema pode atingir num determinado instante. Para os modelos de segurança, cada estado de Markov representa uma combinação diferente de módulos em falha e livres de falha. As transições regulamentam as mudanças de estados que ocorrem dentro de um sistema, em função das taxas de falhas e das taxas de reparo. O fator de cobertura de defeitos pode também ser considerado nesse tipo de modelo.[LAPRIE 91], [SIEWIOREK 74]

Um dos aspectos fundamentais na avaliação dos sistemas que desempenham funções de segurança está na determinação dos Requisitos Gerais de Segurança (**RGS**). Um dos principais métodos na determinação e na verificação dos **RGS** denomina-se Processo de Análise de Risco [LEVESON 86], cujos principais objetivos são:

- examinar os sistemas novos visando identificar perigos em potencial e eliminá-los ou controlá-los;
- examinar sistemas existentes para identificar possíveis perigos, com o intuito de aumentar os níveis de segurança, formular procedimentos de segurança, treinamento de pessoal e aumentar a motivação com relação a uma operação mais eficiente e segura;
- examinar sistemas novos ou existentes, visando demonstrar seus níveis de segurança para as autoridades competentes, de forma a liberar a sua utilização para o público.

Evidentemente, o Processo de Análise de Risco é uma metodologia global, dentro da qual diversas técnicas e ferramentas podem ser utilizadas.

A título de ilustração, nos dois primeiros tópicos desse método, pode-se citar a utilização de ferramentas como árvore de falhas [LEVESON 83], árvore de eventos [ATLEE 93], grafos de causa-efeito, análise dos procedimentos operacionais, análise das interfaces, lista de itens a serem verificados, entre outras.

A partir da determinação dos **RGS**, segue-se a tarefa fundamental da verificação desses requisitos. Dessa forma, são apresentados, a seguir, os princípios básicos de um Programa Global de Avaliação da Segurança de sistemas tolerantes a defeito.

Após a determinação dos **RGS**, inicia-se a tarefa de mapeamento desses requisitos no sistema, com o objetivo de determinar os módulos responsáveis por garantir a segurança do mesmo. A partir desse mapeamento, é possível determinar os componentes de hardware e de software responsáveis pela implementação dos módulos de segurança. Como conseqüência adicional, são obtidos os Requisitos Específicos de Segurança.

Tendo sido definidos os módulos responsáveis pela segurança, inicia-se o processo de análise propriamente dito, contendo:

- análise do Hardware;
- análise do Software;
- análise dos Meios de Detecção dos Defeitos/Erros;
- análise dos Procedimentos Operacionais; e
- modelamento Final da Segurança.

A análise do hardware divide-se basicamente em análise dos módulos redundantes e análise dos módulos "fail-safe". Os módulos redundantes constituem-se naqueles que procuram aumentar a segurança através da redundância, enquanto que os módulos "fail-safe", normalmente os mais críticos, são aqueles que, quando falham, devem levar o sistema para um estado seguro. Uma definição formal do conceito "fail-safe" é apresentada no capítulo 4.

O método de análise dos módulos "fail-safe" engloba as seguintes atividades:

- descrição de funcionamento em operação-normal;
- detalhamento dos requisitos específicos de segurança;

- análise dos modos de falha dos componentes - FMEA;
- análise das entradas impróprias (ruídos, interferências); e
- análise de saídas indevidas.

O método de análise dos módulos de hardware redundantes compõe-se das seguintes atividades:

- análise de cada um dos seus canais com o mesmo procedimento descrito para os módulos "fail-safe"; e
- análise de independência de cada um dos módulos quanto a implementação, fontes de alimentação e sinais de entrada.

Com relação ao software, é conveniente uma avaliação global do mesmo, e não apenas dos módulos identificados como de segurança, em função da grande interdependência característica da sua implementação. Os principais passos nessa avaliação são:

- preparo da documentação de cada rotina (variáveis, descrição funcional, diagrama estruturado, casos de testes, requisito específico de segurança);
- inspeção formal do código-fonte com base num "checklist" para a verificação de prováveis inconsistências decorrentes do uso inadequado da linguagem de programação;
- reuniões formais denominadas "walkthrough" e que têm por finalidade discutir a funcionalidade de todas as rotinas. Essas reuniões não devem passar de 3 horas, contando com a participação de 3 a 4 profissonais. Para cada uma das rotinas, é eleito um defensor, que deverá expor a funcionalidade desta. Durante as reuniões, as rotinas devem ser testadas e realizadas simulações conforme forem as necessidades. Podem ser utilizadas diversas técnicas de teste, como árvore de falhas [KECECIOGLU 91],[LEVESON 91], redes de Petri [COOLAHAN 83], entre outras.
- Vale ressaltar que a técnica de prova de correção pode ser bastante útil em trechos críticos do software, já que a sua utilização ampla em sistemas complexos é bastante dificultosa [RUSHBY 93].
- Evidentemente que, no caso da utilização de métodos formais [FRASER 91],[WILLIAMS 94] no desenvolvimento do software crítico, o enfoque dessa análise

se dará fundamentalmente na verificação da correção e da completeza da especificação formal, além de sua coerência com os demais módulos do sistema. Uma grande preocupação neste tipo de análise recai sobre a verificação do perfeito atendimento da especificação formal em relação aos Requisitos Específicos de Segurança [GERHART1 94], [HALANG 94].

A análise dos meios de detecção e recuperação dos defeitos/erros é uma conseqüência de todas essas atividades, já que a implementação desses meios pode ser realizada por software ou hardware. Determinar esses recursos no sistema é de fundamental importância, pois irá influenciar na adequada utilização de eventuais reconfigurações e, por conseqüência, no modelamento final do sistema.

A análise dos procedimentos operacionais visa identificar seqüências operacionais que possam levar o sistema a um estado inseguro e, portanto, devem ser evitadas ou pelo menos minimizada a sua possibilidade.

O modelamento final da segurança do sistema visa determinar o nível de segurança obtido para o sistema em questão. A medida que melhor representa a segurança de um sistema corresponde ao seu valor de **MTBUF** que, como já visto anteriormente, está relacionado com a função segurança **S(t)** do sistema. Existem técnicas adequadas que auxiliam na determinação desse número, podendo-se citar os modelos combinatórios e de Markov, já comentados anteriormente neste capítulo.

Evidentemente, existem grandes dificuldades para se chegar a um número **MTBUF** realmente representativo do sistema. Dentre essas dificuldades, podemos citar algumas delas:

- dificuldade no modelamento adequado do sistema;
- dificuldade no modelamento adequado do ambiente operacional;
- dificuldades no levantamento das taxas de falhas de componentes mais novos;
- dificuldade na realização de uma avaliação quantitativa do software.

Em função desse panorama e com a crescente complexidade dos sistemas que envolvem aspectos de segurança, deve-se ter a consciência e a responsabilidade de que a obtenção da segurança não se dará apenas em cima de uma única técnica milagrosa, mas fundamentalmente em decorrência de um programa global de segurança que deve estar

presente em todas as fases do ciclo de vida de um sistema [SINGH 90]. A partir dessa preocupação e com o intuito de colaborar no aprimoramento do programa global de segurança, é proposta, no capítulo seguinte deste trabalho, uma metodologia de avaliação da segurança que, em conjunto com outras atividades já apresentadas neste capítulo, deve aumentar ainda mais o grau de segurança dos sistemas existentes, bem como daqueles em projeção.

4. PROPOSTA DE METODOLOGIA DE AVALIAÇÃO DA SEGURANÇA DE SISTEMAS CRÍTICOS

Para uma especificação de requisitos, seja de software, hardware ou sistema, ser considerada robusta, são necessários certos atributos fundamentais: não ser ambígua, ser completa, verificável, consistente, modificável, rastreável e utilizável durante as fases de operação e manutenção.[IEEE - Std 830-1984]

A segurança é definida como uma probabilidade de o sistema não atingir um estado inseguro, que pode afetar vidas humanas e bens materiais. De acordo com diversos pesquisadores dessa área, a análise probabilística da segurança, análise quantitativa, não é muito adequada para assegurar ou avaliar a segurança de um sistema. [LEVESON 83], [HAMLET 92], [BUTLER 93] Esse grupo adota a postura de garantir ou melhorar a segurança através de métodos específicos de projeto.

Em função também da experiência adquirida por diversos engenheiros de segurança de sistemas, concluiu-se que a grande causa de problemas de segurança em sistemas críticos é a existência de uma especificação inadequada de requisitos, ou seja, falta de robustez, além dos erros na aplicação de técnicas de engenharia.[LEVESON 86], [RUSHBY 94], [SHELDON 92]

Pela experiência profissional, também adquirida através de diversos trabalhos de análise de segurança de sistemas críticos, especificamente metro-ferroviários, é possível concluir que grande parte das possíveis falhas inseguras decorrem devidas à falta de especificação com relação ao universo da falha em questão, ou a possíveis mudanças no ambiente operacional.

Em razão dessas observações, pode-se dizer que uma melhoria na análise da completeza de uma especificação deverá aumentar muito a qualidade das mesmas em diversos aspectos, especialmente no que diz respeito à segurança. Dentro desse enfoque, pretende-se desenvolver uma metodologia de avaliação da segurança que tenha, como principal objetivo, a análise de especificações de sistemas críticos, englobando uma avaliação com aspectos informais, sendo complementada com um tratamento formal quando houver uma grande aplicabilidade, usabilidade, inteligibilidade e uma adaptação natural ao processo aplicativo [KNIGHT 94], [FAULK 92]. Nesse sentido, os métodos formais executados rigorosamente podem garantir matematicamente que o modelo está correto com relação a algumas propriedades do sistema, enquanto que o foco principal

da verificação informal pode, então, deslocar-se para as atividades que ainda não foram formalmente especificadas [RUSHBY 94], como por exemplo, a completeza de uma especificação de um sistema crítico em relação ao ambiente de aplicação.

Nessa metodologia de trabalho são utilizados certos símbolos e padrões como elementos de uma "Linguagem", que, em princípio, deve atender as seguintes características básicas:[WILLIAMS 94]

- Notação
- Organização
- Nível de Abstração

A Notação deve ser tal que possa cobrir uma grande variedade de informações, podendo utilizar-se de recursos gráficos, tabulares, simbólicos ou textuais, entre outros. É conveniente evitar o uso de símbolos matemáticos obscuros, que possam acarretar uma não usabilidade e baixa inteligibilidade da respectiva notação.

A Organização é também fundamental, visto que auxiliará no processo de análise, documentação e, evidentemente, de inteligibilidade do sistema. Vale ressaltar que uma das exigências fundamentais de um processo de análise de segurança, independentemente do método a ser utilizado, é o profundo conhecimento da aplicação a ser avaliada. Nesse sentido, a organização dessa metodologia de análise é fundamental para a eficiência do trabalho.

O Nível de Abstração, por sua vez, irá prover meios de se documentar o sistema à medida que o seu entendimento e conhecimento vai se aprofundando. É evidente que a documentação de certos detalhes de projeto serão mais importantes num nível de análise mais avançado do que numa fase de análise de especificação.

Enfim, a Metodologia de Avaliação da Segurança deve ser suficientemente flexível para se adaptar de forma bastante adequada à aplicação, visando melhorar a qualidade dos sistemas desenvolvidos e, fundamentalmente, a sua segurança.

4.1. Metodologia

Este item apresenta uma metodologia de avaliação de segurança, havendo um maior detalhamento na fase de análise da especificação, já apontada como a principal fonte das falhas inseguras em sistemas críticos.

A especificação de sistemas é, em sua maior parte, um conjunto de idéias qualitativas, determinando funções, restrições e abrangência da atuação. A verificação da qualidade dessas especificações, envolvendo principalmente aspectos de segurança, constitui uma tarefa bastante complexa, mas muito motivante e desafiadora. Este trabalho visa, portanto, contribuir com uma maior estruturação no método de análise e avaliação da segurança de especificações de sistemas críticos, especialmente os metro-ferroviários. Nesse sentido, vale novamente ressaltar que a declaração formal dos Requisitos Gerais de Segurança pode e deve auxiliar muito na realização das revisões formais dos "critérios de certificação" contra uma determinada especificação.[RAVN 93]

Em função desse enfoque e dos estudos realizados, sugere-se que:

a. Os métodos a serem utilizados na avaliação de sistemas críticos devem apresentar critérios bem definidos, visando assegurar a completeza da especificação, já que grande parte das falhas nos sistemas é decorrente da omissão da especificação em diversos aspectos.

b. A prova de correção é extremamente útil para verificar e validar critérios e propriedades da especificação, mas ainda não tão eficaz para garantir a completeza da especificação. Para que isso fosse verdadeiro, deveria ser possível formalizar o ambiente da aplicação, tarefa bastante complexa e de tratamento mais adequado, num primeiro momento, através de modelos não tão formais, já que sua inteligibilidade é mais fácil por parte do engenheiro da aplicação. Este, por sua vez, conhece, e muito, o ambiente da aplicação. Num segundo momento, após uma comprovada adequação prática do modelo informal ao ambiente de aplicação, pode-se, então, iniciar o processo de formalização do ambiente em questão.

c. Aplicação de revisões com o intuito de detectar erros na especificação, principalmente relacionados com a falta de completeza.

d. Formalização dos Requisitos Gerais de Segurança, facilitando, dessa forma, as respectivas verificações de atendimento.

e. Utilização de "Computer Aided Software Engineering" (**CASE**) para auxiliar o modelamento da especificação e, dessa forma, permitir a sua simulação, bastante útil num processo de avaliação de segurança. Esses **CASEs** podem,

futuramente, servir de base para a implementação de uma ferramenta de simulação com base no modelo proposto neste trabalho.

Todo esse tratamento é bastante complexo, mas deve auxiliar na melhoria da qualidade dos trabalhos de avaliação de segurança. No entanto, não existe ainda a garantia automática da robustez do sistema e, num aspecto mais específico, de sua segurança, sendo a palavra final ainda dependente do julgamento humano. Isso se deve, principalmente, ao fato da existência de uma lacuna entre a idéia de um sistema e sua formalização e à dificuldade, já mencionada anteriormente, na formalização do ambiente de aplicação. Em razão dessas limitações é que o tratamento através de métodos não tão formais pode auxiliar, num primeiro momento, para a obtenção de sistemas mais seguros [FRASER 91].

Grande parte dos sistemas que envolvem segurança se enquadra na categoria de sistemas de tempo real, por apresentarem as seguintes características:

- aquisição de dados dos sensores
- processamento de dados
- saída para atuadores
- supervisão

Nesses sistemas, a correção não depende apenas da lógica do programa, mas também do instante em que o resultado é produzido. Dessa forma, um dos grandes desafios desses sistemas é o de assegurar o atendimento dos requisitos de temporização [POSPISCHIL 92], [LIN 92]. Sendo assim, um dos principais objetivos é a previsibilidade.

Em relação aos requisitos temporais, a classificação das tarefas envolvidas pode obedecer ao seguinte critério:[STANKOVIC 91]

- "Hard Deadline"
- "Soft Deadline"

Uma tarefa apresenta "hard deadline" quando a mesma deve atender a um requisito de tempo; caso contrário, um estado inseguro pode ser alcançado. Essas tarefas devem ser executadas mesmo na presença de falhas, devendo-se reservar recursos para as mesmas

através da redundância. Estes podem ser um eventual hardware ou software necessários, tais como, espaço em disco ou rotinas de uso exclusivo por determinadas funções. Alguns pesquisadores consideram que a verificação aos requisitos temporais do tipo "hard deadline" são mais facilmente alcançadas através da utilização de software de controle orientado à objeto [GOPINATH 92].

Uma tarefa é classificada como "Soft Deadline" quando, considerada necessária à operação do sistema, degrada o desempenho do mesmo caso não sejam atendidas as suas limitações de tempo, mas não compromete a segurança do sistema, levando-o sempre a uma condição segura.

É compreensível, dessa forma, que, à medida que um sistema vai se tornando mais complexo, a solução através de um tratamento dinâmico e distribuído, em função dos eventos que acontecem no ambiente da aplicação e de sua temporização, torna o sistema mais confiável e seguro do que seria caso se desse uma solução estática ao mesmo [SHIN 92], [STRIGINI 94]. Muitos sistemas apresentam soluções intermediárias, envolvendo soluções estáticas mescladas com algum dinamismo em áreas mais críticas [WENSLEY 78], [ZHAO 87]. Esse fato relaciona-se, por exemplo, com a opção de se utilizarem sistemas operacionais de tempo real, que apresentam um escalonamento de tarefas bastante dinâmico em função de condições externas controladas [WILLIAMS 90], [STANKOVIC 91]. Outra linha de pesquisa nessa área do dinamismo do escalonamento, relaciona-se com a decisão da divisão das tarefas entre os softwares básicos e aplicativos [NATARAJAN 92]. Alguns estudiosos da área entendem que é mais interessante ter o dinamismo no sistema operacional, enquanto outros preferem o dinamismo alocado no software aplicativo, tornando, dessa forma, o sistema operacional menos relacionado com a aplicação.

Assim, a avaliação de segurança de um sistema crítico envolve, obrigatoriamente, não somente o ambiente computacional, como também o ambiente da aplicação, com todas as suas variantes.

Como o esforço, neste trabalho, está mais voltado para a completeza da especificação, por motivos já anteriormente mencionados, deve-se, como primeiro passo, estabelecer critérios que possam auxiliar nessa avaliação. Os critérios devem ser independentes, bastante completos, principalmente no que diz respeito à segurança dos sistemas, englobando todos os grandes subsistemas, desde o Controle, os Sensores e os Atuadores, até o desenvolvimento de um padrão de especificação de alto nível, em

conjunto com os seus procedimentos de análise semântica. Os critérios de avaliação devem abranger diversas categorias de eventos relacionados com variáveis de entrada/saída, estados do sistema, predicados de entrada/saída, relacionamento entre entradas e saídas e transições entre estados. A aplicação desses critérios será efetivada numa especificação de sistema de controle metro-ferroviário, apresentada no capítulo 5, contribuindo, dessa forma, para uma melhor avaliação da segurança dos sistemas a serem desenvolvidos. É evidente que devem fazer parte dessa especificação os denominados Requisitos Gerais de Segurança.

Nesse sentido, os conceitos básicos para a criação desses critérios de completeza constituem os aspectos informais desta metodologia, sendo então formalizados através de uma determinada linguagem. Essa formalização permitirá, assim, a implementação futura de uma ferramenta de apoio.

Obviamente, para a aplicação desta metodologia, a especificação deverá ser representada através de algum padrão, para facilitar o trabalho de verificação dos critérios. O enfoque mais adequado quanto a esse aspecto foi o da utilização da representação do Diagrama de Fluxo de Dados Estendido - (**DFD**) [WARD 86], já que, através dela, os sistemas críticos são representados de maneira adequada e são facilmente inteligíveis, inclusive por parte dos seus usuários finais [SAEED 91]. Outro aspecto a se considerar é que os critérios de completeza serão aplicados sobre um Modelo de Transição de Estados [SHAW 92], em que a transição entre estados se dá pela ocorrência de pré-condições, na forma de "eventos de entrada", e com a realização de "eventos de saida", gerando pós-condições. Nesse sentido, associado a cada **DFD**, é conveniente existir um modelo de transição de estados, sobre o qual haverá a verificação dos critérios de completeza, que, por sua vez, determinarão, como conseqüência, a existência ou não de situações não previstas, que poderão levar o sistema a situações seguras ou inseguras.

O modelo de transição de estados é bastante adequado para essa aplicação, visto que, como já mencionado no capítulo anterior, o modelo adotado para a avaliação da segurança **S(t)** baseia-se no Modelo de Markov, que tem como fundamento o próprio modelo de transição de estados. Assim, o mesmo modelo de transição de estados que está sendo utilizado para verificação dos critérios de completeza pode e deve servir de base para a avaliação global da segurança **S(t)** de um sistema crítico.

Um aspecto adicional importante na utilização deste método de análise é o auxílio na complementação da determinação dos Requisitos Gerais de Segurança quando os

mesmos não foram ainda totalmente definidos ou constituem-se numa tarefa de difícil realização, em função da complexidade do ambiente da aplicação. Nesse caso, esta metodologia serve para orientar o trabalho no sentido de se determinarem os aspectos que podem ser bastante importantes numa especificação de sistemas de segurança. Se a conseqüência da não-verificação de algum critério de avaliação vier a produzir uma condição potencialmente insegura, então tal critério passará também a ser considerado como um Requisito Geral de Segurança.

Os critérios devem ser verificados sobre todos os níveis de especificação. Vale ressaltar novamente que o modelo que será utilizado para verificação dos critérios e representação da especificação é o Modelo de Transição de Estados. Por sua vez, cada predicado de entrada e de saída nesse respectivo modelo será representado através de uma expressão de lógica de predicados, denominada "Well-Formed Formula" (**WFF**).[RICH 83] Os critérios de avaliação da segurança também serão formalizados através de expressões de lógica de predicados **WFF**, acrescidas de funções específicas do modelo de transição de estados e explicadas ao longo da formalização das tabelas de transições de estados e da formlaização dos critérios. Dessa forma, essas expressões **WFF** podem ser transformadas em Forma de Cláusulas, denominadas Forma Normal Conjuntiva, através de regras de conversão. O objetivo final desse formalismo visa tornar bastante inteligível os critérios, além de permitir a sua implementação, bem como a da própria especificação, em regras expressas em linguagem **PROLOG**, que tem como fundamento principal as expressões lógicas. Através desse método, pode-se, futuramente, implementar um protótipo para facilitar os testes da metodologia de avaliação proposta nesta pesquisa. As regras básicas de uma **WFF** e sua conversão em forma de cláusula estão apresentadas no Anexo A dessa tese.

Na realidade, pretende-se, com este trabalho, chegar a sistemas cada vez mais robustos com relação à segurança. A Robustez de um sistema construído a partir de uma especificação depende muito da sua completeza, ou seja, da completeza do ambiente de aplicação adotado, e também da completeza do sistema de controle projetado. Não devem existir eventos observáveis que tornem indeterminado o comportamento esperado do sistema. Dessa forma, o objetivo específico é determinar um conjunto de critérios que, através da verificação da completeza, possam maximizar a perspectiva de se obter um sistema robusto quanto à segurança.

Alguns desses critérios são globais e se aplicam o todos sistemas de controle. Outros são mais específicos e devem ser determinados em função do tipo de aplicação estudado.

4.2 Modelo a ser Utilizado na Representação do Sistema de Controle

Os sistemas de controle modernos, além de manterem e regularem as variáveis, fazem também a supervisão e o planejamento para gerar o escalonamento e a seqüência de eventos na operação do sistema.

O Sistema pode ser representado com as seguintes notações:

- **F** .....Função
- **E**.....Entradas
- **S**.....Saídas
- **T** .....Tempo

O modelo básico a ser utilizado na representação de um sistema está apresentado na figura 4.1.

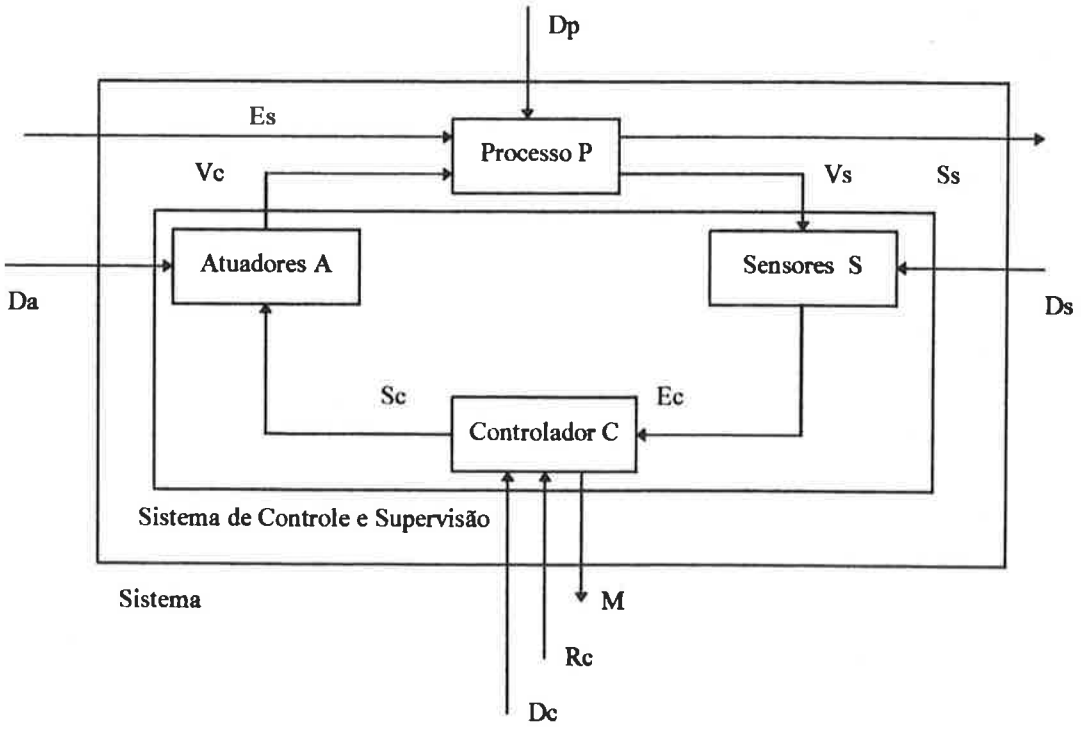


Figura 4.1 - Modelo Básico de um Sistema de Controle e Supervisão

São apresentadas, a seguir, as definições do significado de cada uma das variáveis utilizadas na figura anterior. No capítulo 5, serão exemplificadas essas variáveis para a aplicação específica em um sistema metro-ferroviário.

Es: Domínio de todas as entradas externas do processo sendo controlado, relacionadas com o ambiente da aplicação.

Ss: Domínio de todas as saídas externas do processo sendo controlado, relacionadas com o ambiente da aplicação.

Vc: Domínio de todas as variáveis do processo que são controladas pelo sistema de controle e supervisão.

Vs: Domínio de todas as variáveis do processo que são supervisionadas pelo sistema de controle e supervisão.

Ec: Domínio de todas as entradas recebidas, através dos sensores, pelo Controlador.

Sc: Domínio de todas as saídas enviadas aos atuadores pelo Controlador.

Rc: Domínio de todos os requisitos de comandos dos operadores desse sistema de controle e supervisão recebidos pelo Controlador.

M: Domínio de todas as mensagens, alarmes e avisos, em geral, apresentadas pelo Controlador aos operadores do sistema.

Dp: Domínio de todos os distúrbios externos ao processo sendo controlado, afetando, de alguma forma, o seu comportamento.

Ds: Domínio de todos os distúrbios externos aos sensores, afetando, de alguma forma, o seu funcionamento.

Da: Domínio de todos os distúrbios externos aos atuadores, afetando, de alguma forma, o seu funcionamento.

Dc: Domínio de todos os distúrbios externos ao Controlador e que, de alguma forma, afetam o seu funcionamento.

Com relação ao Processo em si, existe uma dificuldade na sua representação, em função da não linearidade decorrente da sua dinamicidade no tempo. De modo global, pode-se descrever a função do Processo como:

Fp: Vc x Es x Dp x T → Ss x Vs

As responsabilidades pela implementação das funções de controle podem ser distribuídas entre diversos componentes, incluindo dispositivos analógicos, computadores digitais e operadores. Esse fato deve ser considerado na verificação dos critérios quando da análise do nível de detalhamento do projeto correspondente. A função do Controlador pode ser expressa da seguinte maneira:

Fc: Ec x Rc x Dc x T → Sc x M

O Controlador atua no Processo através dos Atuadores, que podem ser representados da seguinte forma:

Fa: Sc x Da x T → Vc

A supervisão do Processo é realizada por meio das informações obtidas pelos Sensores, cujas funções podem ser representadas através da seguinte formalização:

Fs: Vs x Ds x T → Ec

Este modelo apresentado tem como referência inicial o Modelo Clássico, "Feedback Control System", em que não há uma ação corretiva até que as variáveis controladas desviem-se de seu valor esperado. Além disso, nenhuma ação corretiva é realizada até que mudanças nas condições tenham se propagado pela malha de controle. Os atrasos, nesse caso, podem causar sérios danos ao sistema. Existe outro enfoque denominado "Feedforward Control", que apresenta as características não englobadas pelo primeiro, ou seja, há uma ação corretiva antes dos efeitos se propagarem pela malha de controle. Dessa forma, adota-se, neste trabalho, o modelo de sistema com as duas visões de funcionamento, ficando os detalhes desse comportamento embutido no módulo Controlador.

Como a representação se dará através de um modelo de transição de estados, com transições dependentes de eventos de entrada e saída, torna-se fundamental a formalização dos eventos a serem considerados. A representação será através da seguinte padronização: **X↑(t)** - evento da ocorrência da variável **X** no instante **t**, significando uma mudança no valor dessa variável, um recebimento ou uma transmissão da mesma, ou qualquer outro acontecimento relevante relacionado com a aplicação dessa variável. Os eventos representam os sinais de controle no **DFD** estendido. A título de simplificação de notação, esses eventos são representados apenas por **X↑**.

Assim, considerando que **var(D)** representa uma variável do domínio D, tem-se:

var(Es)↑(t): evento de Entrada do Sistema;
var(Ss)↑(t): evento de Saída do Sistema;
var(Dp)↑(t):evento de Distúrbio externo no Processo P;
var(Da)↑(t): evento de Distúrbio externo nos Atuadores A;
var(Ds)↑(t): evento de Distúrbio externo nos Sensores S;
var(Dc)↑(t): evento de Distúrbio externo no Controlador C;
var(Vs)↑(t): evento de Variável Supervisionada;
var(Vc)↑(t): evento de Variável Controlada;
var(Ec)↑(t): evento de Entrada do Controlador C;
var(Sc)↑(t): evento de Saída do Controlador C;
var(Rc)↑(t): evento de Sinal de Requisição de Comando;
var(M)↑(t): evento de Mensagem/Aviso.

Os critérios devem ser aplicados ao "Sistema", ao "Processo", ao "Controlador", aos "Atuadores" e aos "Sensores", visando verificar a sua Completeza, principalmente com relação à segurança, em todos os níveis de especificação.

4.3 Formalização do Modelo de Transição de Estados

Este modelo clássico é formalizado neste trabalho para facilitar a aplicação dos critérios que verificam a completeza de uma especificação, visando, dessa forma, avaliar a segurança do sistema sendo analisado.

Esse modelo é constituído por **estados** e **transições** entre os diversos estados. As transições são rotuladas com expressões lógicas na forma **<Predicado de Entrada // Predicado de Saída>.** Uma transição é realizada se o Predicado de Entrada é avaliado como verdadeiro. Nesse caso, a transição é efetivada, sendo que a determinação das ações a serem realizadas é especificada pelo Predicado de Saída.

O diagrama geral desse modelo é apresentado na figura 4.2, que também pode ser representado através de uma Tabela de Transição de Estados, conforme tabela 4.1.

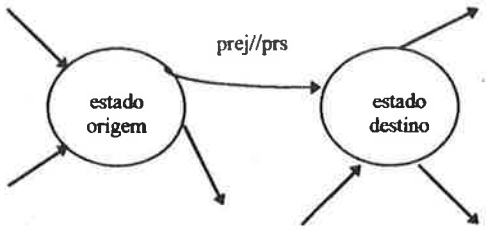


Figura 4.2 - Diagrama de Fluxo de Dados do Modelo de Transição de Estados

	Prei	Predicado de Entrada j - Prej	Prek
Estado n			
Estado Origem		**Predicado de Saída-prs / Estado Destino**	
Estado p			

Tabela 4.1 - Tabela de Transição de Estados

Estando o sistema no **Estado Origem** e sendo verdadeira a avaliação do predicado de entrada **prej** numa transição de saída, então o sistema é transferido para o **Estado Destino** sendo realizada as ações correspondentes ao predicado de saída **prs.**

Esse modelo pode ser representado formalmente por um padrão:

TE : (VARE,VARS,EST,est0,PE,PS,tran,dps), onde:

- **VARE/VARS**: conjunto de Variáveis de Entrada/Saída usadas pelo sistema;
- **EST**: conjunto finito de estados;
- **est0** $\in$ **EST**: Estado inicial no qual o sistema se encontra antes do processo de iniciação;
- **PE**: conjunto de funções booleanas sobre **VARE**, representando os Predicados de Entrada;
- **PS**: conjunto de funções booleanas sobre **VARS**, representando os Predicados de Saída;
- **tran**: função de transição de estado, que define o próximo estado que o sistema irá atingir a partir do estado **est** com a efetivação do predicado de entrada **pr**. **EST x PE** $\rightarrow$ **EST = tran(est,pr); est** $\in$ **EST** $\wedge$ **pr** $\in$ **PE**
- **dps**: Função que determina o predicado de saída quando, o sistema no estado **est**, é efetivado o predicado de entrada **pr**. **EST x PE** $\rightarrow$ **PS = dps(est,pr)** $\wedge$ **est** $\in$ **EST** $\wedge$ **pr** $\in$ **PE**.

A seguir, são apresentadas algumas propriedades desse Modelo de Transição de Estados:

- eventos de entrada são representados pela ocorrência de variáveis de entrada, e eventos de saída são representados pela ocorrência de variáveis de saída;
- os predicados pertencentes aos conjuntos **PE** e **PS** são expressos por operadores booleanos e aritméticos;
- um predicado de entrada **pr** $\in$ **PE** é representado por uma combinação de eventos de entrada e limitações diversas como, por exemplo, verificação de valores aceitáveis, intervalos de tempo, entre outras, dependendo da aplicação analisada;

- um predicado de saída **pr** $\in$ **PS** é representado por uma combinação de eventos de saída e limitações diversas como, por exemplo, verificação de valores aceitáveis, intervalos de tempo, entre outras, dependendo da aplicação sendo analisada;

- o predicado vazio é representado através da notação **null**;

- o valor da variável **x** é expresso através da notação **val(x)**;

- a validade ou consistência da variável **x** é expressa através da notação **validade(x)**;

- a seqüência da ocorrência das variáveis de entrada ou saída é representada através de um índice auxiliar na variável. Dessa forma, $\mathbf{x}_{j-1}\uparrow$ representa uma ocorrência da variável **x** antes de uma outra ocorrência seguinte dessa mesma variável, representada por $\mathbf{x}_j\uparrow$, ou simplesmente $\mathbf{x}\uparrow$, e que indica a última ocorrência dessa variável;

- é necessária a representação de um tempo absoluto e de um tempo relativo a um determinado evento. O tempo absoluto, instante corrente, é simplesmente representado por **t**, enquanto que o instante de um determinado evento é representado por $\mathbf{t(x\uparrow)}$, ou seja, o instante em que ocorreu o evento $\mathbf{x}\uparrow$.

- O relógio absoluto é iniciado quando o sistema recebe o sinal **início**, efetivando-se, portanto, o evento **início**$\uparrow$. Assim, pode-se afirmar que $\mathbf{t(início\uparrow) = 0}$.

4.4 Determinação dos Critérios para a Verificação da Completeza do Sistema

Os Critérios são classificados em função de suas características sobre o Modelo de Transição de Estados e devem ser aplicados conforme o nível de detalhe em que se analisa a especificação, ou seja, nos níveis de sistema, subsistema, módulo ou componente.

Tendo como referência básica o trabalho de [JAFFE 91], e adaptando-se aos aspectos propostos neste trabalho, criaram-se seis categorias básicas de Critérios para verificar a completeza:

- Variáveis de Entrada/Saída;
- Estados;
- Predicados de Entrada;
- Predicados de Saída;
- Relação Entrada/Saída; e

- Transições.

Para melhor entendimento da apresentação dos critérios, adota-se a seguinte padronização:

- **Critério X.X**: apresentação textual do respectivo critério;
- **Definições**: eventuais definições que se façam necessárias, além dos formalismos já apresentados para a descrição do modelo de transição de estados;
- **Notação Formal:** representação formal do respectivo critério.
- **Mensagem de Advertência**: Mensagem informativa ao projetista ou analista do sistema em questão, visando destacar o aspecto em que o sistema não está concordante caso o critério não seja atendido. A partir dessa mensagem, deve-se avaliar a sua **aplicabilidade** e **criticidade** em relação ao ambiente aplicativo.

4.4.1 Critérios para as Variáveis de Entrada/Saída

- **Critério 1.1**: As variáveis de entrada/saída devem ser utilizadas caso contrário, não haveria a necessidade de existirem. Uma variável de entrada qualquer deve ser utilizada em algum predicado de entrada de uma transição. Da mesma forma, uma variável de saída deve ser utilizada em algum predicado de saída de uma transição. Se isso não acontecer, ou a variável de entrada/saída não deve fazer parte do sistema, ou então existe uma omissão na especificação.

Notação Formal:

$\forall$ **ve** $\in$ **VARE , ((pr** $\in$ **PE)** $\wedge$ **(pr(f(ve))).**

$\forall$ **vs** $\in$ **VARS, ((pr** $\in$ **PS)** $\wedge$ **(pr(f(vs))).**

Mensagem de Advertência: A variável **ve/vs** não deve fazer parte do sistema, ou existe uma omissão na especificação quanto à função da variável **ve/vs**.

- **Critério 1.2**: Seja **v** uma variável de entrada ou saída, a **validade(v)** reflete a sua validade ou consistência. Esse atributo pode ser usado, por exemplo, para especificar valores aceitáveis, paridade, precisão das variáveis. Quando do não-

atendimento a essas verificações, uma resposta adequada deve ser declarada, ou, pelo menos, ser armazenada num arquivo histórico, para posterior análise "off-line", conforme for mais conveniente.

Notação Formal:

$\forall$ ve $\in$ VARE, $\forall$ est $\in$ EST, ((pr $\in$PE) $\wedge$
(pr $\equiv$ ($\neg$validade(ve))) $\wedge$ (dps(est,pr) = ps) $\wedge$ (ps = <ação adequada>)).

$\forall$ vs $\in$ VARS, ((pr $\in$PS) $\wedge$ (pr $\equiv$ (validade(vs) $\wedge$ <ação adequada>))).

Mensagem de Advertência: Omissão na especificação quanto à validade da variável **ve/vs**.

4.4.2 Critérios para os Estados

Para uma formalização correta de alguns dos critérios relacionados com os estados, pertencentes ao modelo em questão, será necessária a definição de alguns subconjuntos, bastante importantes neste tipo de avaliação.

Definições:

ESTI : Conjunto dos estados inseguros, ou seja, todos os estados que, mesmo sem a ocorrência de falhas ou entradas impróprias, estão sujeitos a acidentes, envolvendo risco de perdas de vidas humanas e/ou danos materiais de grande valia. Esses estados são conseqüências do não atendimento aos Requisitos Gerais de Segurança.

ESTP : Conjunto dos estados perigosos, ou seja, todos os estados que, com a ocorrência de uma determinada falha ou entrada imprópria, atingem estados considerados inseguros.

ESTSEG : Conjunto dos estados seguros, ou seja, todos os estados que, mesmo com a ocorrência de uma única falha ou entrada imprópria, não atingem nenhum estado inseguro. Nesse aspecto, quando, a partir de um determinado estado, qualquer falha ou entrada imprópria faz com que sempre seja atingido um estado seguro, este é considerado "fail-safe". Se todos os estados de um sistema são "fail-safe", então pode-se considerar o sistema como sendo de segurança intrínseca, ou seja, "fail-safe". Vale ressaltar que essa definição de estado seguro não é totalmente padronizada, visto que

existem outras linhas de trabalho, em que pode ser mais adequado considerar um estado seguro aquele em que, mesmo com duas ou mais falhas ou entradas impróprias, não leve o sistema a atingir nenhum estado inseguro.

ESTSEGD : Conjunto dos estados seguros degradados ou com funções reduzidas. A degradação desses estados é caracterizada pela não habilitação de algumas funções. Esse conjunto de estados constitui-se num subconjunto dos estados seguros **ESTSEG**.

- **Critérios 2.1**: O Sistema deve iniciar e finalizar num estado seguro. O sistema de segurança deve estar apto a funcionar na iniciação e na finalização do sistema, incluindo também, nesse aspecto, a reiniciação após um período fora de operação.

Notação Formal:

$\exists$ **est1** $\in$ **ESTSEG, tran(est0,inicio**$\uparrow$**) = est1.**

$\forall$ **est** $\in$ **EST,** $\exists$ **est1** $\in$ **ESTSEG, tran(est,termina**$\uparrow$**) = est1.**

Mensagem de Advertência: Sistema sendo iniciado ou finalizado num estado não seguro.

Com relação às condições de iniciação, existem duas situações básicas: iniciação após um processo completo de desligamento e após um desligamento temporário, as quais caracterizam a iniciação a frio e a quente, respectivamente. Em ambos os casos, muitos acidentes acontecem em função de um processamento não adequado.

- **Critério 2.2**: O comportamento do sistema com relação às entradas antes da iniciação, após o desligamento e durante o desligamento, deve ser especificado, seja detalhando a forma de utilização dessas entradas, ou podendo ignorá-las de maneira a não comprometer a segurança do sistema.

Definições:

setof(pre,est,PEest) : conjunto dos predicados de entrada de todas as transições de saída do estado **est**, representado por **PEest**.

Notação Formal:

Iniciação

$\forall$ **ve** $\in$ **VE,** $((pr \in PE) \wedge (pr(f(ve)) \equiv inicio\uparrow \wedge ...)) \vee$
$((pr \in PEest1) \wedge (tran(est0,inicio\uparrow) = est1) \wedge pr(f(ve)))$.

Terminação
$\forall$ **est** $\in$ **EST,** $((pr \in PE) \wedge (tran(est,pr) = est0) \wedge (pr(f(ve)) \equiv (termina\uparrow \wedge ...)))$.

Mensagem de Advertência : Especificação omissa quanto à iniciação/reiniciação ou finalização do funcionamento do sistema.

A utilização dos modos de operação auxilia na simplificação da descrição da operação do sistema, permitindo uma visão de alto nível do fluxo de controle do mesmo. Em sistemas de segurança crítica, os modos de operação estão associados a estados operacionais que são subdivididos em estados seguros, estados perigosos e estados inseguros. Os estados perigosos podem ainda ser classificados em estados de alto risco e estados de baixo risco, conforme a probabilidade de serem levados a estados inseguros. Dessa forma, as atitudes a serem tomadas quando se atinge um estado perigoso podem diferir dependendo do modo de operação em que se encontra o sistema.

- **Critério 2.3**: Os caminhos a partir de um estado seguro devem ser especificados. Além disso, deve ser minimizado o tempo de permanência num estado seguro de funções reduzidas, ou seja, um estado em que a degradação é caracterizada pela não habilitação de algumas funções. Deve também ser minimizado o tempo de permanência num estado perigoso. Em alguns casos, a permanência demorada nesses estados pode vir a provocar outras situações inseguras no sistema.

Notação Formal:

$\forall$ **est** $\in$ **ESTSEG,** $\exists$ **est1** $\in$ **EST,** $((pr \in PE) \wedge (pr \neq null) \wedge (tran(est,pr) = est1) \wedge (est1 \neq est))$.

Mensagem de Advertência: Especificação omissa quanto à transição do estado seguro **est**.

Definições:

timeoute(est)↑: evento indicando que o tempo de permanência máxima no estado **est** foi ultrapassado.

Notação Formal:

$\forall$ **est** $\in$ **(ESTSEGD** $\vee$ **ESTP), ((pr** $\in$ **PE)** $\wedge$ **(tran(est,pr) = est1)** $\wedge$
(est1 $\in$ **ESTSEG)** $\wedge$ **(pr** $\equiv$ **(timeout(est)↑** $\wedge$ **...))).**

Mensagem de Advertência: Não está limitado o tempo de permanência num estado seguro de funções reduzidas ou num estado perigoso.

- **Critério 2.4**: Falhas no sistema de segurança devem provocar a transição do sistema para um estado seguro degradado, com o desligamento das funções que envolvem segurança.

Definições:

FSS↑ representa um evento de falha no sistema de segurança.

Notação Formal:

$\forall$ **est** $\in$ **ESTSEG,** $\exists$ **est1** $\in$ **ESTSEGD, tran(est,FSS↑) = est1.**

Mensagem de Advertência: O sistema não está atingindo um estado seguro com a ocorrência do evento de falha **FSS↑** no sistema de segurança.

- **Critério 2.5**: A falha no sistema deve provocar a transição do mesmo para um estado seguro.

Definições:

FS↑ representa o evento de falha no sistema.

Notação Formal:

$\forall$ **est** $\in$ **ESTSEG,** $\exists$ **est1** $\in$ **ESTSEG, tran(est,FS↑) = est1.**

Mensagem de Advertência: O sistema não está atingindo um estado seguro com o evento desta falha **FS↑** no sistema.

4.4.3 Critérios para os Predicados de Entrada

Para que o sistema seja robusto, é necessário também que haja um caminho previsto para todas as entradas em cada estado. Outro aspecto importante, já comentado anteriormente, é a temporização na especificação de sistema críticos, em função do instante da ação. Dessa forma, os critérios para os predicados de entrada estarão organizados na seguinte classificação: aspectos Lógicos, de Temporização e de Capacidade, que são explicados nos seus respectivos itens.

a) Aspectos Lógicos

Os aspectos lógicos relacionam-se com a consistência lógica desses predicados dentro do modelo de transição de estados adotado. Assim, enquadram-se dentro dessa categoria os seguinte critérios:

- **Critério 3.1**: Cada estado deve ter uma transição definida para cada entrada possível.

Notação Formal:

$\forall$ **est** $\in$ **EST,** $\forall$ **ve** $\in$ **VARE,** $\exists$ **est1** $\in$ **EST, ((pr** $\in$ **PE)** $\wedge$ **(tran(est,pr) = est1)** $\wedge$ **pr(f(ve))).**

Mensagem de Advertência: Omissão da especificação no tratamento da variável de entrada **ve**, quando o sistema apresenta-se no estado **est**.

- **Critério 3.2**: O "OR" lógico dos predicados de entrada nas transições de saída de qualquer estado deve constituir uma tautologia; caso contrário, constata-se a presença de alguma situação não prevista no sistema.

Definições:

numberof(pre,PEest): número de predicados de entrada **pre** pertencentes ao conjunto **PEest**;
(i=1 a n) ∨ prei(PEest): operação **OR** de todos os predicados **prei** pertencentes ao conjunto PEest.

Notação Formal:

$\forall$ **est** $\in$ **EST, ((n = numberof(pre,PEest)) $\wedge$ ((i=1 a n) $\vee$ prei(PEest) = T)).**

Mensagem de Advertência: Existe alguma situação em que a especificação está omissa quanto ao comportamento esperado do sistema no estado **est**.

- **Critério 3.3**: Para haver um comportamento determinístico do sistema, é necessário que, num determinado instante, seja verdadeiro apenas um predicado de entrada, correspondente a uma determinada transição a partir do estado corrente do sistema.

Notação Formal:

((prei $\in$ Peest) $\wedge$ (prej $\in$ Peest) $\wedge$ (i $\neq$ j) $\Rightarrow$ ((prei $\wedge$ prej) = F)).

Mensagem de Advertência: A especificação está não determinística no que se refere aos predicados de entrada **prei e prej,** pois ambos são verdadeiros no mesmo instante.

b) Aspectos de Temporização

Os aspectos de temporização englobam as condições relacionadas com os intervalos válidos para as entradas, bem como para a ausência desses sinais. Assim, dentro dessa categoria, enquadram-se os seguintes critérios:

- **Critério 3.4**: Em cada estado o sistema deve ter um comportamento bem definido, ou seja, deve executar alguma transição no caso de não haver entradas por um período de tempo denominado "Time Out". Esse aspecto pode ser também utilizado na reiniciação do sistema quando do não recebimento de determinadas entradas.

Definições:

$(t(ve_i\uparrow) - t(ve_{i-1}\uparrow)) > TM) \Rightarrow timeoutv(ve)\uparrow$.

TM : Tempo máximo de não recebimento de entradas **ve**.

Notação Formal:

$\forall est \in EST, \forall ve \in VARE, \exists est1 \in EST, ((est1 \neq est) \wedge (tran(est,timeout(ve)\uparrow) = est1))$.

Mensagem de Advertência: A especificação está omissa quanto ao comportamento esperado do sistema se não houver a entrada **ve** por um determinado período de tempo quando o sistema está no estado **est**.

- **Critério 3.5**: Todos os eventos de entrada devem ser totalmente limitados no tempo, através da especificação de um tempo mínimo e de um tempo máximo, para sua coerência.

Notação Formal:

$\forall ve \in VARE, (tmin(ve\uparrow) \leq t(ve\uparrow) \leq tmax(ve\uparrow))$.

Mensagem de Advertência: A especificação está omissa quanto ao período de validade da ocorrência da variável de entrada **ve**.

Deve ser especificada uma saída para a ausência do sinal por um período de tempo em relação a algum evento observável. Assim sendo, deve ser estabelecido um tempo específico a partir do qual se inicia a temporização correspondente à ausência de entradas.

- **Critério 3.6**: Para um estado qualquer, deve existir uma transição de saída, cujo predicado de entrada envolva a não existência de uma determinada entrada durante um intervalo específico em relação a algum evento observável.

Definições:

tinício: instante do evento observável;
tduração : período em que deve acontecer **ve**$\uparrow$ em relação ao evento observável;

Notação Formal:

$\forall$ **ve** $\in$ **VARE**, $\forall$ **est** $\in$ **EST**, **((pr** $\in$ **PE)** $\wedge$
(pr $\equiv$ **(t(**$\neg$**ve**$\uparrow$**)** $>$ **(tinício + tduração))).**

Mensagem de Advertência: Especificação omissa quanto ao período de validade da ocorrência da variável **ve** em relação a outros eventos observáveis cujo instante de ocorrência é **tinício.**

c) Aspectos de Capacidade

Embora as entradas fornecidas ao sistema pelo operador ou por um outro sistema dificilmente causem alguma sobrecarga no sistema sendo avaliado, outros problemas, devidos a mal funcionamento, podem causar esse excesso, ultrapassando o limite de capacidade. A robustez, nesse caso, requer que se especifique a forma de se manipularem entradas excessivas e determina um limite de capacidade para tais entradas, como meio de se detectarem possíveis problemas externos e internos ao sistema em questão. Isso implica se determinar o número máximo de entradas dentro de um certo intervalo de tempo. Para os casos em que a capacidade é excedida, deve haver alguma especificação da maneira como o sistema poderá falhar.

Ambos, software e hardware, requerem que uma declaração seja feita sobre o número máximo de entradas **N** por um período de tempo de duração **T.**

A capacidade de entrada **Ce** depende do processo sendo controlado e dos sensores enviando informações sobre o processo. Num sistema de controle, essa capacidade de entrada subdivide seus estados em: estados normais e estados de sobrecarga.

Dessa forma, enquadram-se dentro dessa categoria os seguintes critérios:

- **Critério 3.7**: Uma variável de entrada no sistema requer uma declaração de capacidade.

Notação Formal:

$\forall$ **ve** $\in$ **VARE,** $\exists$ **c, c = Ce(ve).**

Mensagem de Advertência: A especificação está omissa quanto à declaração de capacidade da variável **ve**.

- **Critério 3.8**: Uma declaração de capacidade mínima e máxima deve ser especificada para cada variável de entrada, cuja taxa de chegada não seja limitada por outro tipo de evento.

Definições:

O cálculo de **Ce** do ponto de vista lógico é realizado da seguinte forma:

$t(ve_i\uparrow) - t(ve_{i-n}\uparrow) > T$

$t(ve_i\uparrow) - t(ve_{i-(n-1)}\uparrow) < T \Rightarrow$ **Ce(ve) = n** entradas num período **T**

Notação Formal:

Cemin(ve) $\leq$ **Ce(ve)** $\leq$ **Cemax(ve),** de forma que quando:

Ce(ve) < Cemin(ve) $\Rightarrow$ <desatualização de variáveis>

Ce > Cemax $\Rightarrow$ <sobrecarga>

Mensagem de Advertência: A especificação está omissa quanto aos limites da capacidade lógica de entrada da variável **ve**.

- **Critério 3.9**: Deve ser requerida uma verificação das taxas mínima e máxima de chegada de eventos para cada caminho de comunicação fisicamente distinto. O sistema deve ser capaz de supervisionar o ambiente de comunicação.

Definições:

Nesse caso, **Ce** é calculado do ponto de vista físico, ou seja, para cada caminho físico **cf**.

Notação Formal:

$Cemin(cf) \leq Ce(cf) \leq Cemax(cf)$.

Mensagem de Advertência: A especificação está omissa quanto aos limites da capacidade física de um determinado caminho físico.

- **Critério 3.10**: As respostas às entradas que excedem a capacidade do sistema devem ser especificadas. Essas respostas devem se enquadrar em alguma das classes detalhadas a seguir:

  - Requisitos para gerar mensagens de advertência, ou seja, avisos aos operadores do sistema para adotarem atitudes operacionais que amenizem as conseqüências da sobrecarga.

  - Requisitos para gerar sinais controle para sistemas externos visando a diminuição de capacidade, "slowdown". Nesse caso, a sobrecarga pode ser limitada através de um controle indireto dos sistemas externos. É fundamental a interface de comunicação entre os sistemas envolvidos.

- Requisitos para bloquear os canais em sobrecarga, "lockout", havendo, dessa forma, uma degradação do sistema quanto ao tratamento das informações bloqueadas.

- Requisitos para reduzir a precisão, o tempo de resposta, ou outra característica que permita ao sistema processar uma capacidade maior com relação às entradas. Nesse aspecto, o sistema passa para um estado de degradação, diminuindo a qualidade do processamento das informações de entrada.

- Requisitos para reduzir a funcionalidade do sistema ou, em casos extremos, solicitar o desligamento do controle ou, até mesmo, do processo, quando necessário.

Notação Formal:

$\forall$ **est** $\in$ **EST,** $\forall$ **ve** $\in$ **VE,** $\exists$ **estad** $\in$ **ESTAD, ((pr** $\in$ **PE)** $\wedge$
(pr $\equiv$ **(Ce(ve) > Cemax(ve)))** $\wedge$ **(tran(est,pr) = estad)** $\wedge$ **(dps(est,pr) = prs)** $\wedge$
(prs $\in$ **PS)) :**

a) **prs = < mensagem de advertência>**
b) **prs = < slowdown externo>**
c) **prs = < lockout>**
d) **prs = < slowdown interno>**
e) **prs = < degradação funcional (reconfiguração)>** $\vee$ **< desligamento>**

Mensagem de Advertência: A especificação está omissa quanto à resposta da ultrapassagem do limite máximo da capacidade de entrada da variável **ve** do sistema.

- **Critério 3.11**: Se a degradação ou um processo de reconfiguração é utilizado como meio de atender a uma capacidade excessiva, deve ser especificado um atraso referente à histerese para que o sistema possa retornar ao processamento com carga normal. Esse atraso devido à histerese tem por

objetivo evitar o chaveamento indevido entre carga normal e sobrecarga, também denominado "efeito ping-pong".

Notação Formal:

$$\forall\ \mathbf{est} \in \mathbf{ESTD}, \exists\ \mathbf{est} \in \mathbf{EST}, ((\mathbf{pr} \in \mathbf{PE}) \wedge (\mathbf{tran(estd,pr)} = \mathbf{est}) \wedge$$
$$(\mathbf{pr} \equiv ((\mathbf{t(Ce(ve)} \leq \mathbf{Cemax(ve))} - \mathbf{t(Ce(ve)} > \mathbf{Cemax(ve)))} > \mathbf{H}) \wedge \ldots)).$$

Mensagem de Advertência: A especificação está omissa quanto ao tempo de histerese necessário para o retorno à operação normal, devido à normalidade na capacidade de entrada do sistema referente à variável **ve**.

4.4.4 Critérios para os Predicados de Saída

Para que uma especificação seja robusta em relação aos predicados de saída, deve-se verificar os limites de tempo inferior e superior das saídas. Esses requisitos estão classificados de acordo com a seguinte estrutura: capacidade do ambiente, envelhecimento da informação e latência.

a) Capacidade do Ambiente

Define-se a capacidade do ambiente como sendo a taxa máxima para a qual os atuadores podem aceitar e reagir aos dados produzidos pelo controlador. Esse valor-limite é afetado pelos seguintes elementos:

- limitação de capacidade dos próprios atuadores;
- limitações no comportamento do processo sendo controlado;
- considerações de segurança.

Dentro dessa filosofia, enquadram-se os seguintes critérios:

- **Critério 4.1**: A capacidade do ambiente deve corresponder à capacidade máxima de saída do sistema. Se a capacidade máxima de saída do sistema for excedida, é necessária a realização de alguma ação especial, visando normalizar a taxa de saída.

Definições :

Cs ... capacidade de saída

$\mathbf{t(vs_i\uparrow) - t(vs_{i-n}\uparrow) > T}$
$\mathbf{t(vs_i\uparrow) - t(vs_{i-(n-1)}\uparrow) < T \Rightarrow Cs(vs) = n}$ **num período de tempo T.**

Notação Formal:

$\mathbf{\forall\ est \in EST, \exists\ est1 \in EST, ((pre \in PE) \wedge (tran(est,pre) = est1) \wedge}$
$\mathbf{(pre \equiv (Cs(vs) > Csmax(vs))) \Rightarrow (dps(est,pre) = <ação\ especial>).}$

Mensagem de Advertência: A especificação está omissa quando a capacidade máxima de saída do sistema, em relação à variável **vs**, é excedida.

b) Envelhecimento da Informação

Outro aspecto importante envolve a obsolescência das informações. As decisões de controle devem ser baseadas em dados atualizados do estado do sistema.

- **Critério 4.2**: Todas as entradas utilizadas na especificação de predicados de saída devem ser adequadamente limitadas no tempo, para garantir, dessa forma, o seu não envelhecimento.

Definições:

TV: Tempo de validade de uma entrada, do ponto de vista do não envelhecimento.

Notação Formal:

$\mathbf{((pr \in PS) \wedge pr(f(vs)) \wedge (vs \in VARS) \wedge (vs = f1(ve)) \wedge (ve \in VARE) \Rightarrow}$
$\mathbf{(pr \equiv ((((t(ve\uparrow) + TV) > t(vs\uparrow)) \wedge ...)).}$

Mensagem de Advertência: A especificação está omissa quanto ao envelhecimento da entrada **ve** na produção da saída **vs**.

- **Critério 4.3**: A seqüência de ações perigosas deve ser limitada no tempo, após o qual o sistema deve requerer o cancelamento automático dessas ações e informar ao operador.

  Definições:

  A função **seqtran** está definida formalmente no item 4.4.6 desta tese.

  Notação Formal:

  $\forall$ est $\in$ ESTSEG, ((pr $\in$ PE*) $\wedge$ (seqtran(est,pr) = est1) $\wedge$ (est1 $\in$ ESTP) $\wedge$ (tran(est1,pre) = est) $\wedge$ (pre $\in$ PE) $\wedge$ (pre $\equiv$ ((t - t(est)) > TMAX)) $\wedge$ (dps(est1,pre) = prs)) $\Rightarrow$ (prs = <mensagem ao operador do cancelamento da operação>.

  Mensagem de Advertência: A especificação está omissa quanto à limitação no tempo da ocorrência de uma seqüência de ações perigosas.

c) Latência da Informação

Dado que um sistema de controle não é arbitrariamente rápido, há um intervalo de tempo durante o qual o recebimento de uma nova informação não pode mudar a variável de controle de saída, mesmo que chegue antes dessa geração. Esse intervalo de tempo é influenciado pelo hardware e pelo software do sistema de controle, podendo ser bastante pequeno, mas nunca reduzido a zero. A escolha do sistema operacional a ser utilizado, a lógica de interrupção, a prioridade no escalonamento de tarefas e os diversos parâmetros de projeto serão influenciados pelo valor máximo permitido para esse intervalo de latência.

- **Critério 4.4**: Um fator de latência deve ser incluído na especificação quando uma saída não restritiva é disparada pela ausência ou pela chegada de uma entrada específica, para a qual o limite superior desse intervalo de tempo não é um evento observável simples, em função de prováveis interferências ou da própria dinamicidade do sistema de controle.

Definições:

d: intervalo de tempo de ausência ou de recebimento de uma determinada entrada;

Δt: fator de latência entre o fim do intervalo de tempo **d** e o disparo da saída correspondente a esse evento;

Definições:

restritiva(vs): VARS → Booleana; representa o estado de uma variável ser ou não restritivo em relação às condições de segurança do sistema.

Notação Formal:

$\forall$ **est** $\in$ **EST, ((pre** $\in$ **Peest)** $\wedge$ **(pre** $\equiv$ **(t >** **(t(ve**$\uparrow$**) + d + Δt))) $\wedge$
(dps(est,pre) = prs) $\wedge$ **(prs** $\in$ **PS)** $\wedge$ **(prs(f(vs)))** $\wedge$ **(**$\neg$ **restritiva(vs))).**

Mensagem de Advertência: A especificação está omissa na determinação de um tempo de latência para a ausência ou para o recebimento de um evento de entrada **ve**$\uparrow$, com a correspondente geração de saída **vs**$\uparrow$.

- **Critério 4.5**: Ações contingentes podem ser necessárias na especificação, para tratar os eventos de entrada que ocorrem dentro do período de latência.

Notação Formal:

$\forall$ **est** $\in$ **EST, ((pre** $\in$ **PEest)** $\wedge$
(pre $\equiv$ **(t(ve$_{i-1}$**$\uparrow$**) + d) < t(ve$_i$**$\uparrow$**)** $\leq$ **(t(ve$_{i-1}$**$\uparrow$**) + d + Δt))) $\wedge$ (dps(est,pre) = prs)** $\Rightarrow$
(prs = <ações contingentes>.

Mensagem de Advertência: A especificação está omissa quanto ao comportamento do sistema para os eventos de entrada **ve**$\uparrow$ que ocorrem dentro do período de latência **Δt.**

- **Critério 4.6**: Um fator de latência deve ser especificado para mudanças ocorridas na interface homem-máquina, quando usadas para decisões críticas. Da mesma forma, as ações contingentes devem ser especificadas quando da mudança das informações na interface homem-máquina durante o período de latência. Esse critério está relacionado com o tempo de arrependimento do operador.

Definições:

d: intervalo de tempo entre o recebimento de entradas sucessivas pela interface homem-máquina, para efeitos de decisões críticas pelo operador.

Δt : fator de latência entre o fim do intervalo de tempo **d** e o disparo da saída correspondente.

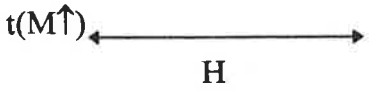
Notação Formal:

$\forall$ **est** $\in$ **EST, ((pre** $\in$ **PEest)** $\wedge$
(pre $\equiv$ **((t(Rc$_{i-1}$**$\uparrow$**) + d) < t(Rc$_i$**$\uparrow$**)** $\leq$ **(t(Rc$_{i-1}$**$\uparrow$**) + d +** Δ**t))** $\wedge$ **(dps(est,pre) = prs)** $\Rightarrow$
(prs = < ações contingentes >).

Mensagem de Advertência: A especificação está omissa na determinação de um tempo de latência **Δt** para mudanças ocorridas na interface homem-máquina, quando utilizadas para decisões críticas, e também omissa nas ações a serem tomadas caso essa mudança ocorra dentro desse período de latência.

- **Critério 4.7**: Um período de histerese deve ser especificado correspondente ao atraso da ação do operador na interpretação das informações a ele apresentadas. Devem ser especificadas as ações caso as informações apresentadas ao operador mudarem durante esse período de histerese.

Definições:



H ...Histerese de tempo de interpretação.

Notação Formal:

$\forall$ est $\in$ EST, ((pre $\in$ PE) $\wedge$ (pre $\equiv$ (t(M_i↑) < (t(M_{i-1}↑) + H))) $\Rightarrow$
((dps(est,pre) = prs) $\wedge$ (prs = <ações contingentes>).

Mensagem de Advertência: A especificação está omissa na determinação de um tempo de histerese correspondente ao atraso na ação do operador na interpretação das informações a ele apresentadas, bem como na especificação das suas ações correspondentes a eventuais mudanças dessas informações durante esse período de histerese.

4.4.5 Critérios para a Relação Entrada/Saída

Existem requisitos que estão relacionados com Entrada/Saída, podendo estes serem classificados de acordo com os seguintes aspectos:

- Capacidade de resposta;
- Espontaneidade.

A Capacidade de resposta e a Espontaneidade lidam com o comportamento do processo e como ele reage às saídas produzidas pelo sistema de controle. Esses parâmetros são supervisionados pela realimentação.

- **Critério 5.1**: Deve haver variáveis supervisionadas que detectem o efeito das variáveis de controle. As informações sobre a característica do processo devem ser utilizadas como características preditivas do comportamento esperado do sistema.

Definições:

VC ....Conjunto de Variáveis de Controle
VS ....Conjunto de Variáveis Supervisionadas

Obs:. Tendo como referência, neste critério, o bloco Controlador, pode-se afirmar que **VC ≡ VARS** e **VS ≡ VARE**.

Notação Formal:

$\forall$ **vc** $\in$ **VC,** $\exists$ **v** $\in$ **VS, v = f(vc).**

Mensagem de Advertência: Não existe, na especificação, variável de supervisão **v** com o objetivo de monitorar a variável de controle **vc**.

- **Critério 5.2**: Para cada variável de controle que possui uma variável de supervisão de comportamento esperado, devem ser avaliadas as condições para resposta muito demorada ou muito rápida.

Definições:

Tmin: tempo mínimo de resposta de supervisão da variável de controle;
Tmax: tempo máximo de resposta de supervisão da variável de controle.

Notação Formal:

$\forall$ **v** $\in$ **VS,** $((v = f(vc)) \wedge (vc \in VC)) \Rightarrow ((pre1, pre2 \in PE) \wedge$
$(pre1 \equiv ((t(vs\uparrow) - t(vc\uparrow)) < Tmin)) \wedge (pre2 \equiv ((t(vs\uparrow) - t(vc\uparrow)) > Tmax)))$.

Mensagem de Advertência: Não existe, na especificação, a determinação de comportamento quando se detecta comportamento fora do esperado da variável de controle **vc**, através da sua monitoração pela variável de supervisão **v**.

- **Critério 5.3**: O recebimento espontâneo de uma entrada, apenas esperada em resposta a alguma saída anterior do sistema, deve ser detectada e respondida como uma situação anormal, realizando alguma ação contingente.

Notação Formal:

$\forall$ **est** $\in$ **EST,** $\forall$ **ve** $\in$ **VARE,** $((ve\uparrow \equiv vs\uparrow \wedge ...) \wedge (vs \in VARS) \Rightarrow$
$((pre \in PE) \wedge (pre \equiv (ve\uparrow \wedge \neg vs\uparrow)) \wedge (dps(est,pre) = prs) \wedge$
(prs = <ações contingentes>)).

Mensagem de Advertência: A especificação do sistema está omissa quanto ao comportamento decorrente do recebimento de uma entrada **ve** não esperada.

4.4.6 Critérios para as Transições

Em particular, os requisitos relacionados com as transições servem para garantir que certos estados sejam alcançáveis.

As características básicas que serão consideradas são Alcance Básico, Comportamento Recorrente, Reversibilidade e Alcance de Estados Seguros.

Para a correta formalização dos critérios relacionados com as transições, serão necessárias as definições de funções adicionais. As funções que serão definidas são **seqtran**, que representa uma seqüência de transições, e a função **coerência,** que representa a veracidade da seqüência de transições dentro de um processamento normal.

Definições:

$\forall$ **est** $\in$ **EST**

$\forall$ **s** $\in$ **PE*** (conjunto de seqüência de Predicados de Entrada), $\forall$ **p** $\in$ **PE** , define-se que:

seqtran : EST x PE* $\rightarrow$ EST;

seqtran(est,sp) = tran(seqtran(est,s),p);

seqtran(est,null) = est;

coerência : PE* $\rightarrow$ Booleana ; representa uma coerência funcional entre os predicados desta seqüência **PE*.** Essa coerência funcional está intimamente relacionada com o ambiente aplicativo.

coerência(sp) = coerência(coerência(s) $\wedge$ p).

coerência(null) = true.

a)Alcance Básico

- **Critério 6.1**: Todos os estados devem ser alcançáveis a partir de um estado inicial. Caso um estado não seja alcançável, há duas possibilidades:

  - o estado não tem função e pode ser eliminado da especificação;
  - o estado deve ser alcançável e a especificação precisa ser modificada.

Notação Formal:

$\forall$ **est** $\in$ **EST, ((pr1** $\in$ **PE*)** $\wedge$ **(seqtran(est0,pr1) = est)).**

Mensagem de Advertência: O estado **est** pode ser eliminado da especificação, ou a especificação precisa ser modificada para incluir tal estado como alcançável.

b)Comportamento Recorrente

- **Critério 6.2**: O comportamento recorrente desejável deve ser parte de um ciclo, ou seja, deve ser possível atingir um estado, de forma coerente, a partir dele mesmo e através de um caminho não vazio, passando por outros estados. Esse critério passa a ter um papel fundamental se o estado em questão for de importância para a segurança do sistema.

Notação Formal:

$\forall$ **est** $\in$ **EST,** $\exists$ **est1** $\in$ **EST, ((pr1** $\in$ **PE*)** $\wedge$ **(pr2** $\in$ **PE)** $\wedge$ **(pr2** $\neq$ **null)** $\wedge$
(tran(est,pr2) = est1) $\wedge$ **(seqtran(est1,pr1) = est)** $\wedge$
(coerência(coerência(pr1) $\wedge$ **pr2))).**

Mensagem de Advertência: O estado **est** da especificação não pode ser atingível a partir dele mesmo por um caminho não nulo e passando por outros estados.

c) Reversibilidade

A reversibilidade está relacionada com a capacidade de os comandos sobre os atuadores serem cancelados ou revertidos por algum outro comando ou combinação.

- **Critério 6.3**: A reversibilidade de um procedimento de operação no sistema por outro procedimento requer a existência de caminhos possíveis entre os estados atingidos e revertidos.

Definições:

operação **ox:** leva o sistema de um estado **esty** para o estado **estx;**

operação **oy**: reversão da operação **ox,** que leva o sistema para o estado **esty** a partir do estado **estx.**

Notação Formal:

$\forall$ **estx, esty** $\in$ **EST, ((ox** $\in$ **PE*)** $\wedge$ **(seqtran(esty,ox) = estx)** $\wedge$ **(coerência(ox))**
$\Rightarrow$ **((oy** $\in$ **PE*)** $\wedge$ **(seqtran(estx,oy) = esty)** $\wedge$ **(coerência(oy))).**

Mensagem de Advertência: Não há reversibilidade da operação **ox.**

d) Alcance dos Estados Seguros

Nesse item, são avaliados os critérios específicos relacionados com os estados seguros **ESTSEG** e as ações esperadas caso os mesmos, por algum motivo, não possam ser alcançados.

- **Critério 6.4**: Não deve haver caminho coerente para um estado perigoso nem para um estado inseguro.

Notação Formal:

$\forall$ est $\in$ EST, $\neg \exists$ estX $\in$ (ESTI $\vee$ ESTP), ((pr $\in$ PE*) $\wedge$ (seqtran(est,pr) = estX) $\wedge$ (coerência(pr))).

Mensagem de Advertência: Há um caminho possível para um estado **estX** perigoso ou inseguro.

Dado que o sistema atingiu um estado perigoso, seja devido a falhas de componentes ou do sistema computacional, erro humano ou distúrbio externo, entre outros, o sistema de controle deverá conduzir o processo para um estado seguro. A decisão da ação adequada a ser tomada deverá depender das condições ambientais em que o processo se localiza no momento.

- **Critério 6.5**: Todo caminho a partir de um estado perigoso deve conduzir a um estado seguro.

Notação Formal:

$\forall$ estp $\in$ ESTP, $\exists$ ests $\in$ ESTS, ((pr $\in$ PE*) $\wedge$ (seqtran(estp,pr) = ests) $\wedge$ (coerência(pr))).

Mensagem de Advertência: Existe um caminho que leva o sistema de um estado perigoso **estp** para um estado não seguro (perigoso ou inseguro).

Pode não ser possível construir um sistema intrinsicamente seguro, ou seja, "fail-safe". Nesse caso, o sistema transformará o estado perigoso em um estado de mínimo risco aceitável, aspecto relacionado com o MTBUF do sistema.

- **Critério 6.6**: Se um estado seguro não puder ser alcançado a partir de um estado perigoso, todos os caminhos a partir desse estado perigoso devem levar o sistema a estados de mínimo risco aceitável.

Definição:

ESTminrisco $\subset$ **ESTP:** conjunto dos estados de mínimo risco aceitável.

Notação Formal:

$\forall$ **estp** $\in$ **ESTP, (pr** $\in$ **PEestp)** $\Rightarrow$ **((tran(estp,pr) = estX)** $\wedge$
(estX $\in$ **(ESTS** $\vee$ **ESTminrisco)))** (está relacionado com o MTBUF do sistema).

Mensagem de Advertência: A especificação está omissa quanto aos estados de mínimo risco aceitável, ou seja, não especifica objetivamente o valor do MTBUF.

5. AVALIAÇÃO DA METODOLOGIA PROPOSTA

Neste capítulo, são apresentadas a evolução dos sistemas de controle metro-ferroviários e a aplicação da metodologia, proposta no capítulo anterior, num sistema modelo de controle metro-ferroviário. São expostos também os resultados obtidos decorrentes da aplicação de cada um dos critérios de verificação da completeza em uma especificação de sistema crítico.

5.1 Evolução dos Sistemas de Controle Metro-Ferroviários

A segurança desempenha um papel cada vez mais importante nos Sistemas de Controle Metro-Ferroviários. Os primeiros sistemas foram consolidados a partir dos conceitos de relés vitais e lógicas a relés, atingindo plenamente a sua finalidade e sendo imunes a vários tipos de interferência, atendendo, dessa forma, seus requisitos gerais de segurança.

A crescente automatização e o aumento da complexidade dos sistemas de controle exigiram uma correspondente evolução dos mecanismos de segurança, até se chegar a utilizar microcomputadores programáveis, tendo sempre como objetivo o aumento da disponibilidade, da confiabilidade e da segurança desses sistemas. Os sistemas de comunicação também estão passando a ter um papel fundamental na segurança dos novos sistemas de controle.[JAEGER-PONNET 93]

Nesse sentido, este item apresenta os conceitos fundamentais e a evolução dos sistemas de controle metro-ferroviários, destacando as diversas arquiteturas utilizadas e as tecnologias empregadas na implantação.

5.1.1 Conceitos Fundamentais

Um sistema de controle metro-ferroviário divide-se basicamente em sistemas de via e sistemas a bordo do trem. Ambos os sistemas têm uma denominação global de "Automatic Train Control", adotado a partir de agora como **ATC**. O **ATC** subdivide-se, por sua vez, em dois subsistemas denominados "Automatic Train Operation" (**ATO**) e "Automatic Train Protection" (**ATP**). O **ATO** é responsável pela operacionalidade do sistema, sendo que o **ATP** desempenha o papel responsável pela implementação das funções de segurança.

O ambiente operacional que é controlado por um sistema de controle metro-ferroviário é composto pelas seguintes entidades:

- trem;
- via; e
- equipamentos de via.

O trem é composto por diversos vagões, podendo variar em função das características operacionais do sistema e da sua tecnologia.

A via é composta normalmente por estações, estacionamentos e zonas de manobra.

Os equipamento de via são constituídos basicamente pelos seguintes dispositivos:

- Equipamento de Trechos de Via

  São equipamentos que pemitem detectar a presença ou não de um trem num determinado trecho de via, fornecendo as indicações de ocupação/desocupação, que são analisadas pelo sistema de controle.

- Máquinas de Chave

  São equipamentos que podem ser comandados pelo sistema de controle, com o objetivo de definir a direção de percurso de um trem. Normalmente, são operadas eletricamente sendo que, em condições de emergência, podem ser controladas através de alavanca manual. As máquinas de chave apresentam duas posições básicas: normal e reversa, determinadas pelo posicionamento de sua ponta de agulha. Conforme apresentado na figura 5.1, que se segue, a posição normal indica não desvio da direção em que o trem se encontrava, enquanto que a posição reversa indica mudança de direção de percurso.

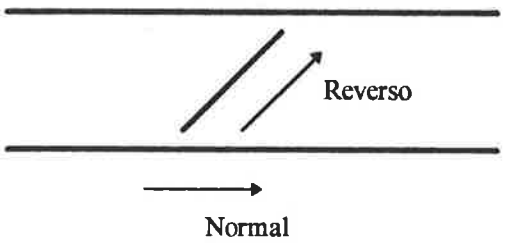


Figura 5.1 - Posicionamento de uma Máquina de Chave

Além desses aspectos, a máquina de chave pode estar travada ou não. O estado de travamento indica não haver possibilidade de movimentação no posicionamento da máquina de chave. Esse travamento é garantido através da não alimentação da energia do motor de deslocamento de posição.

- Sinaleiros

  Os sinaleiros, também denominados "bloqueios", indicam permissão ou proibição de entrada dos trens numa determinada região.

- Dispositivos de Código de Velocidade

  São equipamentos que permitem a transmissão do código de velocidade ao trem num determinado trecho da via.

- Dispositivos de Caixa Quente/Roda Quente

  São equipamentos que auxiliam na determinação, em função da temperatura, de quais eixos dos vagões estão com uma alta possibilidade de rompimento.

- Detector de Descarrilhamento

  Dispositivo colocado na via, para detectar quando há descarrilhamento de algum trem.

5.1.2 Controle de Tráfego Centralizado (**CTC**)

Os sistemas de controle ferroviários mais convencionais apresentam uma arquitetura denominada Controle de Tráfego Centralizado (**CTC**). Nesse tipo de sistema, os blocos de segurança da sinalização ferroviária ficam distribuídos ao longo da via, sendo

implementadas, inicialmente, com o uso de réles vitais. Esses blocos de segurança, também denominados de "intertravamentos vitais", interligam-se através de cabos de comunicação, permanecendo conectados também a um centro de controle, que realiza as operações básicas da ferrovia, ou seja, controle da movimentação dos trens. Esse tipo de arquitetura tem apresentado uma grande evolução com relação à tecnologia, como por exemplo, a implantação de intertravamentos vitais à base de microprocessadores.[EGNOT 93], [RAO 87], [VERNAZZA 90] Além dessa característica, há uma grande expansão na funcionalidade dos centro de controle, incluindo tarefas de maior apoio à operação, à manutenção e também ao gerenciamento da ferrovia.

Todos esses fatores têm tornado esses sistemas de controle cada vez mais complexos e automáticos, provocando a criação de novas arquiteturas e novas tecnologias que melhor atendam às necessidades crescentes da sociedade.[MÜNCH 93], [TAKAOKA 91]

5.1.3 Sistema Avançado de Controle de Trens (**ATCS**)

A Associação Americana das Ferrovias (**AAR**), em conjunto com a Associação Canadense das Ferrovias (**ACR**), trabalharam, em 1984, na definição de um Sistema Avançado de Controle de Trens, denominado **ATCS**, com o objetivo de aumentar a segurança, eficiência e economia das operações da ferrovia.[COLL 90]

Os trens são controlados por uma liberação de movimento, enquanto que os veículos auxiliares, para apoio à manutenção, são controlados por permissão de ocupação de trechos da via. Quando necessário, a violação do limite da liberação de movimento gera uma aplicação automática de freio pelo **ATCS**.

Os grandes blocos do sistema **ATCS** são: Sistema de Despacho, Sistema de Comunicação, Equipamento da Locomotiva, Equipamento de Veículo Auxiliar e Equipamento à Margem da Via.

O Sistema de Despacho é responsável pelo planejamento da movimentação dos trens, monitoração de estado da cada movimentação e registro dos dados operacionais. Os principais elementos são o console de despacho, o sistema de gerenciamento, o sistema de segurança, a interface com o sistema de comunicação e a interface com o sistema de informação de gerência corporativa.[KADONO 91]

O Sistema de Comunicação consiste de uma rede de nós e enlaces para prover comunicação entre o despachador e as locomotivas, os veículos auxiliares e os dispositvos à margem da via. O enlace de comunicação entre a locomotiva e o despachante é considerado vital, sendo necessária a criação de um protocolo adequado para tal fim. Esse enlace é considerado vital, pois um dos aspectos funcionais do sistema é permitir o despacho do trem através de dados enviados via rádio. Como o sistema de comunicação deve ser muito confiável, devem ser estudados alguns aspectos, como regiões de sombra de comunicação e interferência eletromagnética.[SHEIKH 90] Com relação a esse sistema, pode-se destacar o início da utilização do satélite **Inmarsat** e do sistema "Global Position System" (**GPS**) na função de rastreamento dos trens.[HOPE 93]

O equipamento da locomotiva compreende um rádio, um computador de bordo, "display" da locomotiva, controle de freio, odômetro e interrogador de "transponder". O interrogador irradia um sinal de rádio que está acoplado indutivamente com o "transponder", que está localizado na via. Através desse sinal o "transponder" é energizado, permitindo que este transmita seu código de informação de volta ao interrogador. Essas informações auxiliam na localização precisa do trem ao longo da ferrovia. O computador a bordo da locomotiva executa, entre outras funções, a restrição de liberação de movimento e o gerenciamento de diversos itens relacionados com a manutenção da locomotiva.

Os veículos auxiliares na manutenção, comunicam-se com o despachante através de um terminal de bordo contendo uma antena de rádio freqüência (RF) e um computador auxiliar.

O equipamento à margem da via consiste de uma unidade de interface com a via, de dispositivos controláveis e não controláveis. A unidade de interface com a via estabelece a comunicação entre os dispositivos de campo e o sistema de comunicação do **ATCS** através de um pacote de comunicação. Fazem parte dos dispositivos controlados as máquinas de chave e as cancelas dos cruzamentos. Dentre os dispositivos não controlados, tem-se os transponders, detetores de violação das máquinas de chave, detetores de roda quente e caixa quente e detetores de descarrilhamento.

Antes que um trem deixe o terminal origem, um arquivo com o plano de viagem é transmitido do despachante para a locomotiva. Esses dados permitem ao computador de bordo da locomotiva calcular as características de frenagem do trem e predizer sua

velocidade esperada em cada ponto de sua viagem. Em seguida, o despachante estabelece a autorização de movimento desde que a rota seja validada pelo sistema de segurança. A medida que o trem percorre a rota, o computador de bordo calcula a posição do trem comparando-a com a posição fornecida pelos transponders e efetuando eventuais correções. Em função das posições determinam-se também os limites de autorização de movimento, restrição de velocidade, entre outras funções. Quando o maquinista não age corretamente em caso de ultrapassagem do limite de velocidade, o computador de bordo inicia o processo de frenagem, comunicando tal evento ao sistema de despacho.

5.1.4 Sistema Europeu de Controle de Trem (**ETCS**)

Diferente da característica das ferrovias norte-americanas, que se destacam pela extensão, os sistemas ferroviários europeus caracterizam-se por apresentar distâncias menores, grande quantidade de pátios e estações e grande variedade de tipos de sistemas.

Visando uma maior interoperabilidade na sinalização e no controle das ferrovias européias, profissionais europeus iniciaram o projeto de um sistema padrão de controle metro-ferroviário. Existem atualmente na Europa 16 (dezesseis) tipos de sistemas de controle que devem ser integrados numa rede ferroviária internacional. Essa operação integrada engloba também sistemas de tarifação, informação ao passageiro e treinamento da operação e manutenção. Todo esse trabalho está sendo financiado pela União Européia e denomina-se **ERTMS** - Sistema de Gerenciamento de Tráfego Ferroviário Europeu. O **ERTMS** compõe-se de diversos subsistemas, sendo o principal denominado "European Train Control System" (**ETCS**).[RAILWAYS 94] Devido às diversidades de arquiteturas e de tecnologia, o programa **ETCS** está dividido em três grandes projetos: "Eurocab", "Eurobalise" e "Euroradio". O "Eurocab" descreve os equipamentos de bordo do trem. O "Eurobalise" corresponde ao sistema de transmissão de dados na via, e o "Euroradio" engloba o sistema de transmissão de dados de sinalização, via rádio, para o trem.

O "Eurocab" deverá ser compatível com os sistemas **ATCs** existentes. Para atingir esse objetivo, ele é baseado no conceito de Via de Dados, sobre a qual dispositivos de entrada e saída serão conectados através de interfaces adequadas.[KUSAKARI 91] Dentro do "Eurocab", localiza-se o principal processador do **ETCS**, que executa a lógica de controle e de segurança, enfatizando, dessa forma, a tendência de

deslocamento da inteligência de controle: da via para o trem. Também conectado a essa via de dados, apresenta-se um outro processador que, entre outras funções, armazena o perfil da rota com a finalidade de calcular as distâncias de frenagem, processa as informações dos tacômetros, registra eventos importantes e realiza o controle de portas. A informação que está sendo utilizada nessa via de dados é padronizada para independer do sistema de origem ou de destino.

À medida que o trem percorre a rota, o transponder ou os loops indutivos do sistema "Eurobalise" enviam ao trem um pacote de informações. Estas são colocadas na via de dados para serem recebidas pelo processador do **ETCS**.

Da mesma forma que os outros sistemas do **ETCS,** o "Euroradio" possuirá transmissores e receptores com interface própria com a via de dados. Esse sistema irá transportar informações vitais ao trem, como código de velocidade e liberação de movimento.

A implantação do **ETCS** se dará em duas fases em função da existência de diversos sistemas **ATC** de bordo e de via. Na primeira fase, deve ser interligado o sistema "Eurocab" com os **ATC**s de via já existentes na Europa. Numa segunda fase, o sistema "Eurocab" já deve estar interligado com a nova infra-estrutura englobando o "Eurobalise" e o "Euroradio".

5.1.5 Sistema de Controle por Blocos Móveis

Os sistemas de blocos móveis [WEIR 92] surgiram em função da necessidade crescente, em sistemas metroviários, de um "headway" cada vez menor tanto em horários de pico como de baixa utilização, que corresponde ao intervalo médio de tempo entre trens. Além desses fatores, deve-se considerar também os custos envolvidos com o trem formado por diversos vagões e com a construção das estações. O peso desses fatores pode diminuir consideravelmente com a utilização da tecnologia dos blocos móveis.

Num sistema metroviário, o fluxo de passageiros pode ser fundamentalmente afetado por dois fatores: capacidade dos trens e freqüência de partidas, que está relacionado com o headway do sistema.

Os dispositivos de via tradicionais, utilizados para detectar ocupação/desocupação de trechos de via e também utilizados para transmitir código de velocidade ao trem, denominam-se "circuitos de via", e são projetados para um headway mínimo em torno de

90 segundos. Esses números estão relacionados com um trem de 6 a 8 vagões de 23 metros de comprimento cada um, transportando, cada vagão, 270 passageiros, numa proporção de 4 passageiros/m^2.

O sistema de blocos móveis abre a possiblidade da operação com intervalos menores entre composições de menor tamanho. A primeira experiência desse sistema aconteceu em 1985 no minimetrô de Vancouver, seguida por outra instalação, em 1989, numa linha de baixa capacidade do metrô de Paris.

O princípio básico que sustenta essa nova tecnologia, ainda emergente, fundamenta-se na possibilidade da dinamização do tamanho do dispositivo equivalente ao circuito de via tradicional. Essa nova implementação abrange desde guias de onda ao longo da via, com pequenos furos periódicos através dos quais se realiza a comunicação com a composição, até a utilização de pequenos laços indutivos, através dos quais as informações de posição e velocidade são transmitidas. Através dessa nova filosofia de sinalização, pode ser possível, em horários de pico, o estabelecimento de trechos de via de ocupação/desocupação menores, compatíveis com as distâncias de frenagem e margens de segurança, que permitirão, em última instância, uma maior quantidade de trens na via, diminuindo, conseqüentemente, o headway final do sistema. Nos horários de menor movimento, a velocidade comercial dos trens pode ser aumentada e facilitado o seu controle através do aumento do tamanho dos trechos de via de ocupação/desocupação.

5.2 Aplicação do Modelo em Sistemas de Controle Metro-Ferroviários

Pode-se observar pela apresentação feita nos itens anteriores que a complexidade dos sistemas de controle metro-ferroviários tem aumentado drasticamente nestes últimos 10 (dez) anos, sempre procurando atender às necessidades crescentes dos usuários desse importante meio de transporte.

Evidentemente, em paralelo com toda essa evolução, deve-se ter um cuidado especial com os aspectos de segurança envolvidos na aplicação desses sistemas.

Assim, é apresentada, neste item, a aplicação prática do modelo proposto no capítulo anterior em um Sistema Modelo de Controle Metro-Ferroviário, com o intuito de se avaliar sua praticidade, eficiência e benefícios, avaliando os resultados obtidos e destacando certas considerações fundamentais na avaliação da segurança de um sistema

de controle metro-ferroviário. É apresentada, no Anexo B deste trabalho, uma lista exemplo contendo alguns Requisitos Gerais de Segurança (**RGS**) aplicáveis ao sistema de controle metro-ferroviário, bastante importante num trabalho de avaliação da segurança.

5.2.1 O Sistema de Controle Metro-Ferroviário Adotado

Para efeito de uma avaliação mais compreensível com relação ao sistema metro-ferroviário, são realizadas algumas simplificações, do ponto de vista funcional, visando destacar os aspectos mais importantes. Nesse sentido, será avaliado um sistema de controle de via e, não, a bordo do trem. Além disso, é enfocado, para efeito de avaliação, somente o bloco Controlador, por apresentar uma maior complexidade e cobertura funcional.

Conforme apresentado no capítulo anterior, o bloco Controlador **C** é constituído pelas seguintes entradas e saídas:

- **Ec**: Entradas do Controlador;
- **Sc**: Saídas do Controlador;
- **Rc**: Requisições de Comando;
- **M**: Mensagens, Alarmes e Avisos ao operador; e
- **Dc**: Distúrbios Externos ao Controlador.

Para o bloco Controlador **C**, as variáveis de entrada englobam as Entradas do Controlador **Ec**, as Requisições de Comando **Rc** e os Distúrbios Externos **Dc**. Por outor lado, as variáveis de saída englobam as Saídas do Controlador **Sc** e as Mensagens ao Operador **M**.

As Entradas do Controlador (**Ec**) a serem consideradas neste sistema modelo são:

- Ocupação/Desocupação do trechos de via : **ocp**;

  Essa informação vinda da via indica se um determinado trecho de via está ocupado ou desocupado por um trem.

- Estado das Máquinas de Chave; normal ou reverso (**pos**) e travada (**tr**);

A informação de posicionamento da máquina de chave indica se a ponta de agulha dessa máquina de chave está na posição normal ou na posição reversa. A informação de travamento informa se a máquina de chave está ou não travada, não possibilitando nenhum movimento da mesma.

- Indicação dos aspectos dos Sinaleiros: aberto (**sina**) e fechado (**sinf**);

  Quando o sinaleiro apresenta o aspecto amarelo, indica liberação de percurso de rota pelo trem; no entanto, quando o aspecto é vermelho, a indicação é de proibição de percurso da rota.

- Indicação do Código de Velocidade (**cv**);

  Essa informação corresponde ao código de velocidade que está sendo requisitado para o trem num determinado trecho da via.

As Saídas do Controlador (**Sc**) a serem consideradas neste sistema são:

- Comando para Máquina de chave; normal ou reverso (**cmmch**) e travada (**trmch**);

  Esse sinal comanda a máquina de chave para a posição normal ou para a posição reversa, conforme for solicitado. O comando de travamento coloca a máquina numa situação que não é possível a sua movimentação.

- Comando de Geração de Código de Velocidade (**cmcv**);

  Esse comando envia, a um determinado trecho da via, um código de velocidade a ser transmitido ao trem quando ele estiver percorrendo aquela região.

- Comando de Acionamento de Sinaleiro: abertura (**cmas**) e fechamento (**cmfs**);

  Esses sinais de comando têm o objetivo de colocar o sinaleiro num estado de permissão de percurso de uma rota, ou seja, sinaleiro aberto, ou de proibição de percurso da rota, com o sinaleiro fechado.

As Requisicões de Comando (**Rc**) a serem consideradas neste sistema são:

- **inicia**;

  Essa requisição tem por objetivo iniciar o funcionamento do sistema.

- **termina**;

  Essa requisição tem por objetivo o desligamento do funcionamento do sistema.

- Requisição para Alinhamento de Rota (**rar**);

  Essa requisição tem por finalidade o alinhamento de uma rota a ser percorrida pelo trem. A rota é caracterizada por um percurso, que pode ser formado por diversos trechos de via, máquinas de chave, um sinaleiro de origem e um sinaleiro de destino. Neste modelo de sistema adotado, serão consideradas apenas as rotas em condições normais de operação, ou seja, a rota a ser percorrida está desocupada. No entanto, vale lembrar que existem rotas denominadas de chamada, que são alinhadas mesmo com a ocupação de algum trecho dentro da mesma. Esse tipo de rota é muito utilizado, por exemplo, para rebocar trens quebrados na via.

- Requisição para Cancelamento de Alinhamento de Rota (**rcar**);

  Essa requisição existe para permitir ao operador cancelar a sua requisição de alinhamento efetuada anteriormente. Normalmente, existe um intervalo de tempo, a partir da requisição de alinhamento, dentro do qual a requisição de cancelamento é aceita; caso contrário, ela não é contemplada pelo sistema.

- Requisição para Proibição de Alinhamento de Rota (**rpar**);

  Essa requisição é bastante útil quando, por algum motivo, for necessário o impedimento do percurso de determinados trechos da via pelo trem, como por exemplo, em regiões em manutenção. Quando essa requisição é efetuada, ela tem prioridade sobre os demais comandos, por se tratar de uma ação restritiva ao sistema. Nesse sentido, após a sua requisição, as rotas já alinhadas entrarão em processo de cancelamento.

- Requisição para Cancelamento de Proibição de Alinhamento de Rota (**rcpar**);

  Essa requisição tem o objetivo de retirar a proibição alocada ao sistema através de uma requisição anterior de proibição. Ela é bastante utilizada quando o motivo pelo qual justificou-se a utilização de uma proibição já não existe mais. No entanto, deve-se ter muito cuidado nessa requisição, pois uma ação indevida pode levar o sistema a situações perigosas e inseguras.

Os Distúrbios Externos ao Controlador **Dc** têm sua origem nas interferências eletromagnéticas, eletrostáticas e correntes de fuga, podendo afetar o funcionamento esperado do sistema.

Dentre as diversas mensagens **M** ao operador, pode-se citar, como exemplo:

- confirmação de requisições de comando solicitadas;
- avisos de mensagens de erro;
- avisos para efetuar manutenção;
- histórico de eventos ocorridos; e
- indicação do estado da via.

Essas informações do bloco Controlador **C** podem ser representadas num diagrama de fluxo de dados simplificado mostrado na figura 5.2. Este diagrama apresenta uma bolha funcional global, que representa as funções a serem realizadas pelo Controlador, e uma bolha de controle, que engloba todos os sinais de controle envolvidos na comunicação com a bolha funcional. Os sinais de controle são apresentados na figura de maneira geral, para facilitar o entendimento, já que a quantidade dos sinais de controle é bastante grande, e detalhada nas tabelas de transição de estados apresentadas a seguir.

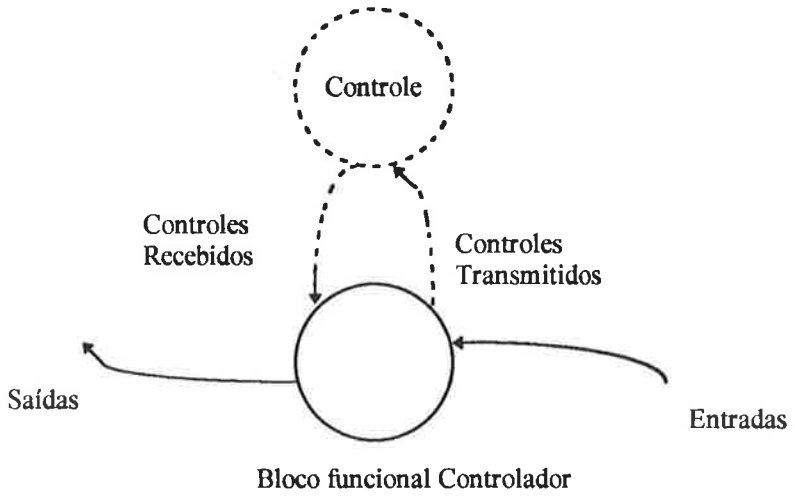


Figura 5.2 - Diagrama de Fluxo de Dados Esquemático do Bloco Controlador

Os sinais de controle, trocados entre a bolha de controle e a bolha funcional, esquematizadas na figura 5.2, são apresentados nas tabelas 5.1 e 5.2.

A tabela 5.1 contém os sinais de controle recebidos pela bolha de controle, denominados também de Predicados de Entrada, com as suas respectivas variáveis de entrada utilizadas.

A tabela 5.2, por sua vez, apresenta os sinais de controle transmitidos pela bolha de controle, denominados de Predicados de Saída, juntamente com as respectivas variáveis de saída afetadas.

Predicados de Entrada	Variáveis de Entrada
inicia↑	inicia
fiminicia↑	ocp,tr,pos,sinf,sina,cv;
termina↑	termina
fimtermina↑	ocp,tr,pos,sinf,sina,cv;
rar↑	rar
alinhamentoOK↑	ocp;tr
alinhamentonãoOK↑	ocp;tr
travamentoOK↑	ocp;tr;pos
travamentonãoOK↑	ocp;tr;pos
ocp(1°Trecho)↑	ocp
ocp(últimotrecho)↑	ocp
rcar↑ ∧ timeout↑	rcar
timeoutcanc↑	rcar
rpar↑	rpar
rcpar↑	rcpar
ocp(trechointerm)↑	ocp
perigosuperado↑	ocp;cv;tr;pos;sinf;sina
liberação↑	ocp;cv;tr;pos;sinf;sina

Tabela 5.1- Tabela de Predicados de Entrada

Predicados de Saída	Variáveis de Saída
ativainic↑	trmch;cmcv;cmfs
ativacontrol↑	-
ativaterm↑	trmch;cmcv;cmfs
desativacontrol↑	-
ativaalinhamento↑	-
cmmch↑∧trmch↑	cmmch trmch
cmas↑ cmcv↑	cmas;cmcv
cmfs↑ ∧ cmcv↑ ∧ ativacancaut↑	cmfs;cmcv;trmch
cmfs↑ ∧ cmcv↑ ∧ ativacanc↑	cmfs;cmcv
M↑ cmcv↑ cmfs↑	cmcv;cmfs
M↑	M

Tabela 5.2 - Tabela de Predicados de Saída

As tabelas de transição de estados, que representam o funcionamento deste sistema modelo, são apresentadas através das tabelas 5.3 a 5.7. Nessas tabelas, está detalhado o comportamento do sistema em questão em função dos eventos de entrada ocorridos. A explicação detalhada dessas tabelas se dará ao longo da verificação dos critérios apresentados no item 5.2.2.

	inicia↑	fiminicia↑	termina↑	fimtermina↑
est 0	ativainic↑/est1	null/est0	null/est0	null/est0
est 1	null/est1	ativacontrol↑/est2	ativaterm↑/est3	ativaterm↑/est3
est 2	null/est2	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 3	null/est3	ativaterm↑/est9	ativaterm↑/est3	destivacontrol↑/est0
est 4	null/est4	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 5	null/est5	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 6	null/est6	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 7	null/est7	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 8	null/est8	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3
est 9	null/est9	ativaterm↑/est9	ativaterm↑/est3	ativaterm↑/est3

Tabela 5.3 - Tabela de Transição de Estados

	rar↑	alinhamentoOK↑	alinhamento nãoOK↑	travamentoOK↑
est 0	null/est0	null/est0	null/est0	null/est0
est 1	null/est1	null/est1	null/est1	null/est1
est 2	ativaalinhamento↑/ est4	null/est2	null/est2	null/est2
est 3	null/est3	null/est3	M↑/est3	null/est3
est 4	-/est4	cmmch↑∧trmch↑/ est4	M↑/est2	cmas↑∧cmcv↑/ est2
est 5	null/est5	M↑/est5	M↑/est5	M↑/est5
est 6	null/est5	M↑/est6	M↑/est6	M↑/est6
est 7	null/est7	M↑/est7	M↑/est7	M↑/est7
est 8	null/est8	M↑/est8	M↑/est8	M↑/est8
est 9	null/est9	M↑/est9	M↑/est9	M↑/est9

Tabela 5.4 - Tabela de Transição de Estados

	travamentonãoOK↑	ocp(1°Trecho)↑	ocp(últimotrecho)↑	rcar↑∧timeout↑
est 0	null/est0	null/est0	null/est0	null/est0
est 1	null/est1	M↑∧cmcv↑∧ cmfs↑/est8	M↑∧cmcv↑∧ cmfs↑/est8	null/est1
est 2	null/est2	cmfs↑∧cmcv↑∧ ativacancaut↑/est5	M↑∧cmcv↑∧ cmfs↑/est8	ativacanc↑∧ cmfs↑∧cmcv↑/est6
est 3	null/est3	M↑∧cmcv↑∧ cmfs↑/est8	M↑∧cmcv↑∧ cmfs↑/est8	null/est3
est 4	M↑/est2	M↑∧cmcv↑∧ cmfs↑/est8	M↑∧cmcv↑∧ cmfs↑/est8	M↑/est2
est 5	M↑∧cmcv↑∧ cmfs↑/est8	null/est5	null/est2	M↑/est5
est 6	M↑∧cmcv↑∧ cmfs↑/est8	M↑/est5	M↑∧cmcv↑∧ cmfs↑/est8	M↑/est6
est 7	null/est7	M↑∧cmcv↑∧ cmfs↑/est8	M↑∧cmcv↑∧ cmfs↑/est8	M↑/est7
est 8	null/est8	null/est8	null/est8	null/est8
est 9	null/est8	null/est8	null/est8	null/est9

Tabela 5.5 - Tabela de Transição de Estados

	timeoutcanc↑	rpar↑	rcpar↑	ocp(trechointerm)↑
est 0	null/est0	null/est0	null/est0	null/est0
est 1	null/est1	null/est1	null/est1	M↑∧cmcv↑∧cmfs↑/est8
est 2	null/est2	cmfs↑∧cmcv↑∧M↑/est7	M↑/est2	M↑∧cmcv↑∧cmfs↑/est8
est 3	null/est3	null/est3	null/est3	M↑∧cmcv↑∧cmfs↑/est8
est 4	M↑/est4	cmfs↑∧cmcv↑∧M↑/est7	M↑/est4	M↑∧cmcv↑∧cmfs↑/est8
est 5	M↑/est5	cmfs↑∧cmcv↑∧M↑/est7	M↑/est5	null/est5
est 6	M↑/est2	cmfs↑∧cmcv↑∧M↑/est7	M↑/est6	M↑∧cmcv↑∧cmfs↑/est8
est 7	M↑/est7	null/est7	M↑/est2	M↑∧cmcv↑∧cmfs↑/est8
est 8	null/est8	null/est8	null/est8	null/est8
est 9	null/est9	null/est9	null/est9	null/est8

Tabela 5.6 - Tabela de Transição de Estados

	perigosuperado↑	liberação↑
est 0	null/est0	null/est0
est 1	null/est1	null/est1
est 2	null/est9	null/est2
est 3	null/est3	null/est3
est 4	null/est9	null/est4
est 5	null/est9	null/est5
est 6	null/est9	null/est6
est 7	null/est9	null/est7
est 8	M↑/est9	null/est8
est 9	null/est9	M↑/est2

Tabela 5.7 - Tabela de Transição de Estados

Para uma melhor compreensão das tabelas de transição de estados anteriormente apresentadas, será explicado, a seguir, o significado de cada um dos estados presentes. Explicações mais detalhadas a respeito das tabelas de transição de estados se darão ao longo dos textos explicativos sobre as verificações dos critérios, apresentadas no item 5.2.2 deste trabalho.

- **est0** - Estado de Repouso: Nesse estado, o sistema está desligado, sem nenhuma função de controle ou supervisão ativada.

- **est1** - Estado de Iniciação: Nesse estado, o sistema está em processo de iniciação, atualizando variáveis da via, iniciando variáveis de controle e ativando as funções de controle e supervisão do sistema.

- **est2** - Estado de Espera: Esse estado corresponde ao sistema estar com suas funções de controle e supervisão ativadas, aguardando a ocorrência de alguma requisição por parte do operador, algum evento disparado pelo bloco Controlador ou algum evento de entrada vindo da via.

- **est3** - Estado de Terminação: Nesse estado, o sistema está em processo de desligamento, desativando as funções de controle e supervisão, mas também verificando as condições da via e do bloco Controlador.

- **est4** - Estado de Alinhamento: Esse estado é decorrente da solicitação do alinhamento de uma rota. Nele são realizadas as verificações necessárias para poder ser liberada uma rota para um trem. As primeiras verificações relacionam-se com a não ocupação dos trechos de via envolvidos na rota e com o não travamento das máquinas de chave. Se alguma máquina de chave envolvida na rota a ser alinhada já estiver travada, isso indica que já existe outra rota alinhada e liberada utilizando essa máquina de chave, não podendo ser alinhada uma outra rota. Em seguida, após essa verificação preliminar de permissão de alinhamento, são enviados comandos à via, para posicionamento e travamento das máquinas de chave. Após a confirmação da efetivação desses comandos, a rota é então liberada para ser percorrida pelo trem. Essa liberação se efetiva através da abertura do sinaleiro envolvido e da geração de código de velocidade no trecho de via a ser percorrido.

- **est5** - Estado de Cancelamento Automático: O sistema entra nesse estado quando o trem invadiu o primeiro trecho da via que faz parte da rota alinhada. Nesse estado, à medida que o trem percorre a rota em questão, os trechos de via já percorridos são liberados para outras rotas, bem como as máquinas de chave já percorridas são destravadas, podendo ser utilizadas em outras rotas. Quando o trem atingir o último trecho de via da rota alinhada, o sistema é transferido, então, para o estado de espera **est2.**

- **est6** - Estado de Cancelamento: O sistema entra nesse estado quando é solicitado pelo operador o cancelamento de uma rota já alinhada ou em alinhamento. Evidentemente existe um intervalo de tempo, contado a partir do pedido de alinhamento da rota, durante o qual é possível solicitar o seu cancelamento. Após a aceitação da requisição de cancelamento pelo sistema, o sinaleiro é fechado e colocado código de velocidade zero no trecho de via na entrada da rota. Em seguida, deve ser aguardado um determinado tempo para certificar-se que o trem realmente não invadiu a rota alinhada. Se houver a invasão da rota antes do término desse tempo, o sistema deixa de permanecer nesse estado e passa para o estado de cancelamento automático, **est5.** Se o trem não invadir a rota nesse período de tempo, a rota é liberada para outra requisições e o sistema é transferido para o estado de espera, **est2.**

- **est7** - Estado de Proibição de Alinhamento: O sistema é conduzido para esse estado quando uma requisição de proibição de alinhamento de rota é recebida. Por se tratar de uma requisição de restrição, ela tem preferência sobre todos as outras condições do sistema. Nesse estado, nenhuma rota pode ser alinhada no trecho em questão, e o sistema só deixa esse estado se o mesmo receber do operador uma requisição de cancelamento de proibição de alinhamento de rota. Normalmente, esse estado ocorre quando existem condições na via que exigem a parada do fluxo de trem, como por exemplo, região da via em manutenção.

- **est8** - Estado Perigoso/Inseguro: Evidentemente, o sistema não é conduzido a esse estado espontaneamente. Essa transição acontece no sistema, aqui adotado como modelo, quando algum evento não coerente com a situação de controle e situação operacional acontece. Nesse caso, é originada uma condição perigosa/insegura, que deve ser detectada pelo sistema para ser adequadamente tratada e, se possível, retirar o sistema dessa situação. Dependendo das condições que o sistema se encontre, esse estado pode ser considerado uma situação perigosa, enquanto que, em outros casos, pode ser considerado uma situação já insegura, conforme definição apresentada no capítulo 4. Neste modelo, não foi explicitamente considerado o efeito das falhas de hardware, mas apenas problemas de eventos não esperados, provavelmente decorrentes de alguma outra falha, seja da movimentação dos trens, seja de entradas impróprias realizadas pelo operador, ou seja, de falhas no software ou no hardware. Evidentemente, à medida que se avança no nível de detalhe do sistema, as análises das falhas de hardware passam a ter uma grande influência, realizadas através do método FMEA, já comentado no capítulo 3 deste trabalho.

- **est9** - Estado Seguro Degradado: Esse estado é atingido pelo sistema quando, após detectar a presença de um estado perigoso/inseguro, o sistema consegue transferir o sistema para um estado em que há uma degradação funcional, porém caracterizando-se num estado seguro. Nesse estado, conforme forem as condições na via, o sistema passa a ter uma degradação nas suas condições operacionais, condição necessária para ele atingir, naquele instante, as condições de segurança exigíveis. Esse estado é também atingido em função de eventos não esperados e que não levam o sistema obrigatoriamente a um estado perigoso/inseguro.

5.2.2 Verificação dos Critérios

Este item apresenta a verificação de todos os critérios de completeza especificados no capítulo 4 deste trabalho. Para cada um dos critérios, serão feitas avaliações no sentido de se veirificar ou não o seu atendimento em relação ao modelo de sistema adotado. Ao longo dessas avaliações, são apresentadas importantes observações, que devem auxiliar na melhoria da segurança dos sistemas de controle metro-ferroviários. A formalização de cada critério apresentada no capítulo 4 auxiliou fundamentalmente numa maior precisão e objetividade dos mesmos, facilitando, dessa forma, as avaliações sobre as tabelas de transição de estados consideradas neste modelo.

Critério 1.1:

Conforme é observado pela tabela 5.1, todas as variáveis de entrada são utilizadas em pelo menos um predicado de entrada. De forma análoga e de acordo com a tabela 5.2, todas as variáveis de saída são utilizadas em pelo menos um predicado de saída.

Critério 1.2:

Com relação às variáveis de entrada do Contralador (**Ec**), pode-se dizer que deve ser verificada a validade com relação ao valor do código de velocidade na via. Nesse sentido, se o código enviado ou recebido não corresponde a um código válido, esse fato caracteriza uma situação incoerente e, portanto, o sistema deve ser colocado no estado est8, gerando o seguinte predicado de saída **(cmcv↑ ∧ cmfs↑ ∧ trmch↑ ∧ M↑)**. Esse predicado de saída corresponde a ações, na região em que houve o problema, de limitações de velocidade, fechamento de sinaleiros e travamento das máquinas de chave, além de mensagens específicas ao operador. Quando houver falta de consistência nas informações de ocupação/desocupação dos trechos de via, os mesmos devem ser considerados como ocupados, prevalecendo, dessa forma, a condição de segurança. No caso de falta de consistência no posicionamento e travamento das máquinas de chave, o sistema deve considerar as máquinas de chave sem posicionamento definido e sem travamento, contribuindo, assim, para a segurança do sistema. Na eventualidade de falta de consistência nas informações dos sinaleiros, os mesmos devem ser considerados no aspecto de impedimento de movimentação de trens.

Com relação às requisições de comando por parte do operador, devem ser efetuadas verificações de consistência para cada uma dessas requisições. Se houver problemas na

consistência dessas requisições, o processamento do sistema deve ignorar tal evento de entrada, fornecendo mensagem de aviso ao operador.

Antes do envio de cada comando de saída **(cmmch,trmch,cmcv,cmas,cmfs)** para a via, devem ser verificadas as suas respectivas consistências. Caso não haja uma correta verificação, deve ser suspensa a ação do respectivo comando.

Qualquer um dos estados de falta de consistência anteriormente apresentados constituem-se em estados de alguma forma de degradação do sistema, pertencentes ao conjunto **ESTSEGD**.

Critério 2.1:

O sistema deve iniciar as variáveis de entrada **Ec** num estado seguro, ou seja, **ocp** deve indicar ocupação, **pos** não deve corresponder a nenhum posicionamento, **tr** deve indicar não travamento das máquinas de chave, os sinaleiros devem ser considerados fechados **sinf,** e os códigos de velocidade **cv**, iguais a zero.

Com relação aos comandos de saída **Sc** e às requisições de comando **Rc**, devem ser iniciados em estado de não acionamento, ou não ativação.

Todas essas considerações são necessárias para que o estado **est1** seja considerado seguro.

No processo de terminação, o sistema deve ser levado a um estado seguro respeitando as mesmas condições das variáveis anteriormente descritas. Em função dessa condição, é que o sistema é transferido para o estado **est3** quando recebe o evento **fimtermina↑** num instante incoerente com o processo. O sistema só é conduzido ao estado **est0** de repouso depois de receber o evento **fimtermina↑**, já estando no estado de terminação **est3**.

Critério 2.2:

O comportamento do sistema com relação às entradas antes da iniciação, após o desligamento e durante o desligamento, está especificado na tabela de transição de estado, sempre tendo como referência a segurança do sistema.

Depois que o sistema encerra o processo de iniciação, atualizando corretamente as variáveis, evento este sinalizado através do evento **fiminicia↑**, ele é transferido do

estado **est1** para o estado **est2** de espera, em que permanece até que novos eventos aconteçam. Qualquer que seja o estado do sistema, exceto os estados **est0** e **est1**, ao receber o evento **fiminicia**↑, este faz com que o sistema seja transferido para o estado seguro degradado **est9**, para efeito de segurança, pois se trata de um evento não coerente, indicando a presença de alguma falha no sistema de controle e supervisão.

O processo de terminação é provocado pelo evento **termina**↑, levando o sistema para o estado de terminação **est3**, exceto se ele estava no estado de repouso **est0.** O sistema deixa esse estado de terminação **est3** ao receber o evento **fimtermina**↑, indicando que as variáveis supervisionadas e controladas foram colocadas no estado seguro. As características desses estados seguros já foram comentadas no critério 1.2.

Critério 2.3:

As tabelas 5.3 a 5.7 especificam os caminhos a partir de qualquer estado, inclusive os estados seguros. Essas mesmas tabelas apresentam os estados **est7** e **est9,** que caracterizam-se por serem estados seguros degradados, e o estado **est8,** que caracteriza-se por ser um estado perigoso. Para todos esses estados, deve ser especificado um tempo máximo de permanência, visando evitar levar o sistema para uma condição insegura. Esse tempo deve ser determinado em função das condições operacionais do sistema e também do ambiente em que localiza o processo, como por exemplo, perfil do público atendido, existência ou não de túneis, existência ou não de fogo, entre outros fatores a serem considerados. Assim, um trem parado por muito tempo dentro de um túnel pode vir a provocar situações perigosas e inseguras. Outro exemplo, a título de ilustração, é a permanência, em demasia, de um trecho de via no estado de proibição de alinhamento de rota **est7,** podendo provocar sérios problemas na disponibilidade global do sistema e, por conseqüência, originar problemas de segurança.

Critério 2.4 e Critério 2.5:

O tratamento adequado para as falhas em qualquer parte do sistema ou no sistema de segurança não está detalhado nas tabelas de transição de estados, mas vale ressaltar a importância da análise de todos os modos de falhas do sistema de segurança com o respectivo estado conseqüente. Esse estado conseqüente deve, no mínimo, fazer parte dos estados seguros degradados, **ESTSEGD**. Com relação aos demais blocos do sistema não responsáveis pela segurança, é importante destacar a necessidade de uma análise de interferência, visando verificar que uma falha em um bloco qualquer não provoque uma situação perigosa ou insegura no sistema. Evidentemente, todos os modos de falhas

devem ser contemplados no modelamento final do sistema e, por conseqüência, no cálculo final do seu MTBUF.

Vale ressaltar aqui que os eventos não esperados nas tabelas de transição de estados são tratados no sentido de transferir o sistema para o estado **est8** ou **est9**. Esses eventos não esperados podem ser uma conseqüência de alguma falha em um bloco do sistema.

Critério 3.1:

De acordo com as tabelas 5.3 a 5.7 de transição de estados, para todas as entradas possíveis, existe uma transição definida em cada estado do sistema.

Critério 3.2:

Para cada estado do sistema, estão previstas transições de saída para todos os predicados de entrada. No entanto, podem existir situações em que nenhum dos predicados em questão é verdadeiro, pelo menos por algum tempo, ficando, nesse caso, o sistema no estado em que se encontra. Dessa forma, deverá ser determinada a necessidade ou não de um tempo máximo de permanência num estado, em função de sua funcionalidade e dos aspectos de segurança envolvidos. Esse aspecto será tratado, em detalhe, no critério 3.4.

Critério 3.3:

Observando os diversos predicados de entrada nos estados do sistema, conclui-se que podem existir vários predicados verdadeiros no mesmo instante. Para contornar esse não determinismo, é necessário estabelecer prioridades ou alguma outra forma de ordem de escalonamento. Do ponto de vista de segurança, é fundamental dar prioridade para os predicados de entrada mais restritivos. Essa definição pode variar em função do estado em que o sistema se encontra. Assim, em todo estado que existir uma transição para um estado de detecção de perigo, e o predicado de entrada responsável por essa transição acontecer no mesmo instante que um outro predicado de entrada, deve-se dar prioridade para o primeiro predicado, permitindo, com isso, que o sistema de controle fique consciente da situação de perigo que o mesmo se encontra. Através desse raciocínio, os predicados de entrada das transições para os estados **est7**, **est8** e **est9** devem ter prioridades no tempo em relação aos demais predicados de entrada. Da mesma forma, os comandos restritivos (**rcar** e **rpar**) devem ter prioridade no tempo sobre os demais predicados que não se encaixam na situação descrita anteriormente.

Critério 3.4:

A permanência exagerada num determinado estado pode indicar ou não a existência de problemas no sistema. Assim, é conveniente um estudo do tempo de permanência em cada estado do sistema.

- **est0**: Não há condições de controle, pois o sistema está desligado;

- **est1**: Se o sistema permanecer muito tempo neste estado, indica algum problema na iniciação do sistema. Dessa forma, deve-se estabelecer um tempo máximo neste estado em função das tarefas necessárias a serem realizadas na iniciação. Se esse tempo for ultrapassado, pode-se avisar o operador do sistema e transferi-lo para o estado **est9**, com comandos restritivos em campo.

- **est2**: Se o sistema permanecer muito neste estado, indica a inexistência de requisições por parte do operador, além da inexistência de eventos na via. Em função das condições dinâmicas da via, como por exemplo, o headway esperado, o sistema pode ser transferido, com a ativação de comandos restritivos, para o estado **est9**, após um determinado período de tempo, visando garantir uma maior segurança para o mesmo.

- **est3**: A permanência excessiva do sistema nesse estado indica um tempo exagerado no processo de terminação, podendo o sistema, nesse caso, ser transferido parta o estado **est9**, para efeito de segurança, com a ativação de comandos restritivos.

- **est4**: A presença do sistema em demasia neste estado indica problemas no processo de verificação das condições de alinhamento e travamento de uma rota, como ocupações dos trechos de via envolvidos e posicionamento e travamento das máquinas de chave. Nesse aspecto, é adequado transferir o sistema para o estado **est9**, para efeito de segurança, avisando também o operador dessa ocorrência e acionando comandos restritivos.

- **est5**: A permanência exagerada do sistema neste estado indica problemas no cancelamento automático de rotas, como por exemplo, demora do trem em atingir o seu destino especificado. Nesse caso, é conveniente avisar o operador, sendo o sistema transferido para o estado **est9**, para garantir a sua segurança, ativando comandos restritivos.

- **est6**: O tempo exagerado de permanência neste estado indica problemas na temporização de espera de invasão da rota, usualmente em torno de 60 segundos.

Dessa forma, é conveniente a transferência do sistema para o estado **est9,** com o acionamento de comandos restritivos.

- **est7**: Este estado caracteriza-se por ser degradado, pois existem restrições operacionais no sitema devido a algum problema existente. Assim, a permanência máxima neste estado deve levar em consideração as condições técnicas do sistema e as condições do ambiente operacional, entre elas o headway, o perfil dos passageiros, a existência de túneis, a existência de fogo, entre outros aspectos. Caso sejam ultrapassados os tempos máximos estipulados, o sistema deve ser transferido para o estado **est9**.

- **est8**: Este estado caracteriza-se fundamentalmente pela existência de uma condição perigosa que deve ser tratada adequadamente. Nesse sentido, o tempo máximo de permanência neste estado deve ser função das condições técnicas do sistema e das condições do ambiente de operação. Evidentemente quanto maior o tempo de permanência do sistema neste estado, menor será o valor do seu MTBUF.

- **est9**: Este estado caracteriza-se por ser um estado seguro, com restrições operacionais na via devido a problemas operacionais e de manutenção. Dessa forma, deve-se estabelecer um tempo máximo, a fim de que a permanência exagerada neste estado não provoque outras situações perigosas no sistema. Os aspectos que devem ser considerados aqui estão também relacionados com as condições técnicas e do ambiente de operação e manutenção. Vale ressaltar que todos os tempos que estão relacionados com as condições técnicas do sistema e com as condições do ambiente de operação e manutenção estão diretamente relacionados com o MTTR, que mede também a eficiência da manutenção corretiva.

Critério 3.5:

Para a verificação deste critério, deve-se analisar todos os eventos de entrada, que são, na realidade, uma conseqüência de variáveis de entrada. Dessa forma, devem ser verificadas as temporizações das variáveis de entrada:

- **ocp↑**: a ocorrência deste evento deve ser limitada por um tempo mínimo e um tempo máximo, para efeito de validade da sua informação. O tempo mínimo está relacionado com um período mínimo necessário de processamento do sistema em questão para atualizar tal variável, ou seja, dependente do tempo de varredura da via e do processamento necessário. O tempo máximo, por sua vez, está relacionado com o

ambiente operacional, ou seja, com o período máximo permitido para não ser considerada envelhecida a informação de ocupação/desocupação. Evidentemente, à medida que se aumenta a velocidade comercial dos trens, esse tempo máximo deve ser diminuído, tendo sempre em mente o não envelhecimento da informação. Esse tempo também será dependente do comprimento do trecho de via para efeitos de ocupação/desocupação.

- **pos↑,tr↑,sina↑,sinf↑,cv↑:** Em todos estes casos, valem os mesmos comentários feitos anteriormente e que garantem o não envelhecimento das informações utilizadas pelo sistema.

Com relação às requisições de comando, estas não precisam ser limitadas no tempo, a não ser a Requisição de Cancelamento de Alinhamento de Rota (**rcar**), que deve acontecer dentro de um tempo máximo a partir da requisição de alinhamento de rota (**rar**). Trata-se de um arrependimento. Esse tempo pode variar em função das condições operacionais, estando normalmente em torno de 5 a 10 segundos.

O não atendimento a esse critério deve fazer com que o sistema considere as informações em questão no seu sentido mais restritivo, ou seja:

- **ocp**: trecho de via ocupado;
- **pos**: não posicionamento da máquina de chave;
- **tr**: não travamento da máquina de chave;
- **sina, sinf**: sinaleiro fechado;
- **cv**: código de velocidade zero.

Critério 3.6:

Como em todos os estados estão sendo avaliados todos os predicados de entrada do sistema, basta então verificar se existe algum desses predicados de entrada que deva acontecer em um intervalo de tempo a partir da ocorrência de um outro evento de entrada. Essa avaliação é feita tomando por base todos os predicados de entrada apresentados na tabela 5.1.

- **inicia↑:** não depende de outro predicado de entrada;

- **fiminicia↑:** depende do predicado de entrada **inicia↑,** mas já verificado no critério 3.4, estado **est1**;

- **termina↑:** não depende de outro predicado de entrada;

- **fimtermina↑:** depende do predicado de entrada **termina↑,** mas já verificado no critério 3.4, estado **est3**;

- **rar↑:** não depende de outro predicado de entrada;

- **alinhamentoOK↑:** depende do predicado de entrada **rar↑;** dessa forma, deve ser especificado um tempo máximo **tduração** a partir da ocorrência do evento **rar↑** para a observação do evento **alinhamentoOK↑** ou **alinhamentonãoOK↑.** Esse tempo deve estar adequado ao tempo de processamento esperado para a tarefa de alinhamento de rota. No término desse período, o sistema deve ser transferido para o estado de espera **est2**, avisando ao operador da não aprovação do alinhamento da rota;

- **alinhamentonãoOK↑:** mesmo comentário anterior;

- **travamentoOK↑:** depende do predicado **alinhamentoOK↑,** pois o travamento da rota só é realizado após a verificação do seu alinhamento. Assim, deve ser especificado um intervalo de tempo **tduração,** contado a partir da ocorrência do evento **alinhamentoOK↑,** para a observação do evento **travamentoOK↑** ou do evento **travamentonãoOK↑.** Esse tempo está relacionado com a duração da tarefa de travamento de uma rota já alinhada. No término desse período de tempo, o sistema deve ser transferido para o estado **est9**, para efeito de segurança, apresentando mensagem de indicação do fato ao operador;

- **travamentonãoOK↑:** mesmo comentário anterior;

- **ocp(1°Trecho)↑:** depende de outro evento de entrada quando for requisitado o alinhamento de uma rota. Nesse caso, deve ser especificado um intervalo de tempo **tduração** a partir da ocorrência do evento **travamentoOK↑,** quando o sinaleiro correspondente à liberação da rota é aberto e o código de velocidade permite esse deslocamento, estando o sistema no estado **est2**. Geralmente, esse tempo corresponde a 60 segundos. No término desse período, o sistema deve ser transferido para o estado **est6,** correspondente, nesse caso, ao cancelamento de uma rota por demora da invasão da mesma pelo trem;

Quando for pedido o cancelamento de uma rota, a ocupação do seu primeiro trecho é um evento não esperado, não se encaixando, portanto, na avaliação desse critério;

- **ocp(últimotrecho)↑:** depende da ocorrência do evento **ocp(1°Trecho)↑,** já tendo sido verificado no critério 3.4, estado **est5**;

- **rcar↑ ∧ ¬timeout↑:** evidentemente, esse predicado de entrada ocorrerá se houver um pedido de cancelamento de alinhamento de rota e não houve a ultrapassagem do tempo máximo de arrependimento para tal pedido a partir da ocorrência do evento **rar↑.** Caso haja a ocorrência do evento **rcar↑** e já foi ultrapassado o tempo máximo de arrependimento, o sistema deve ignorar tal pedido, permanecendo no estado em que se encontrava;

- **timeoutcanc↑:** esse evento depende da ocorrência de um pedido de cancelamento **rcar↑,** já verificado no critério 3.4, estado **est6**;

- **rpar↑:** não depende de outro evento de entrada;

- **rcpar↑:** não depende de outro evento de entrada;

- **ocp(trechointerm)↑:** depende da ocupação do primeiro trecho de uma rota alinhada e do perfil de velocidade do trem. No entanto, só será verificado o período de tempo para ser alcançado o último trecho da rota. A implementação dessa verificação é necessária em casos de Rota de Chamada e na implementação da função rastreamento de trens, não considerados neste exemplo de aplicação;

- **perigosuperado↑:** não depende de um evento de entrada específico;

- **liberação↑:** não depende de um evento de entrada específico;

- **sina↑, sinf↑ e cv↑:** dependem dos eventos **travamentoOK↑, travamentonãoOK↑, rcar↑, rpar↑, rcpar↑, ocp(1°Trecho)↑ e timeoutcanc↑.**

  No caso dos eventos **travamentoOK↑** e **rcpar↑,** se após um determinado tempo **tduração,** dependente dos atrasos envolvidos no sistema, não acontecer o evento **sina↑** ou **cv↑,** isso indica que o comando para abrir o sinaleiro para a rota alinhada e travada ou o comando de cancelamento de proibição de alinhamento de rota não foram efetivados. Nesses casos, deve-se enviar mensagem ao operador e transferir o sistema para o estado **est9**.

No caso dos eventos **rcar↑, rpar↑, travamentonãoOK↑, ocp(1°Trecho)↑** e **timeoutcanc↑**, se após um determinado tempo **tduração**, dependente dos atrasos envolvidos no sistema, não acontecer o evento **sinf↑** ou **cv↑**, isso indica que a requisição para cancelamento de alinhamento de rota ou para a proibição de alinhamento de rota não foi efetivada. Dessa forma, o operador do sistema deve ser avisado e o sistema, transferido para o estado **est8**, pois caracteriza-se uma situação perigosa.

Critério 3.7:

A necessidade de declaração de capacidade deve ser verificada para cada variável de entrada do sistema:

- **ocp**: a capacidade está relacionada com a taxa de varredura, na via, desta informação;
- **pos**: a capacidade está relacionada com a taxa de varredura, na via, desta informação;
- **tr**: a capacidade está relacionada com a taxa de varredura, na via, desta informação;
- **sina,sinf**: a capacidade está relacionada com a taxa de varredura, na via, desta informação;
- **cv**: a capacidade está relacionada com a taxa de varredura, na via, desta informação;
- **inicia**: não requer declaração de capacidade;
- **termina**: não requer declaração de capacidade;
- Requisição para Alinhamento de Rota (**rar**): a capacidade depende da absorção pelo sistema de uma quantidade máxima de requisições desse tipo;
- Requisição para Cancelamento de Alinhamento de Rota (**rcar**): a capacidade depende da absorção pelo sistema de uma quantidade máxima de requisições desse tipo;
- Requisição para Proibição de Alinhamento de Rota (**rpar**): a capacidade depende da absorção pelo sistema de uma quantidade máxima de requisições desse tipo;
- Requisição para Cancelamento de Proibição de Alinhamento de Rota (**rcpar**): a capacidade depende da absorção pelo sistema de uma quantidade máxima de requisições desse tipo.

Critério 3.8:

Para cada variável de entrada que deva existir uma declaração de capacidade, será avaliada a necessidade de um valor mínimo e de um valor máximo:

- **ocp**: a grande importância neste item é com relação à capacidade mínima, pois está relacionada com o não envelhecimento da informação. Assim, essa capacidade irá depender do número total de ocupações a serem atualizadas num determinado intervalo de tempo. Pode-se dizer que o tempo de não envelhecimento da informação irá depender fundamentalmente da velocidade comercial do trem e do tamanho mínimo do trecho de via. Por outro lado, a capacidade máxima está ligada com a capacidade interna de processamento do sistema e dos sensores envolvidos;
- **pos**: mesmo comentário anterior;
- **tr**: mesmo comentário anterior;
- **sina, sinf**: mesmo comentário anterior;
- **cv**: mesmo comentário anterior;
- Requisição para Alinhamento de Rota (**rar**): a capacidade mínima está relacionada com a necessidade operacional do sistema, dependendo em muito da sua complexidade e do grau de abrangência. A capacidade máxima está relacionada com a capacidade de processamento do sistema;
- Requisição para Cancelamento de Alinhamento de Rota (**rcar**): mesmo comentário anterior;
- Requisição para Proibição de Alinhamento de Rota (**rpar**): mesmo comentário anterior;
- Requisição para Cancelamento de Proibição de Alinhamento de Rota (**rcpar**): mesmo comentário anterior.

Critério 3.9:

Para cada caminho físico distinto que recebe um determinado tipo de informação, o sistema deve ser capaz de supervisionar sua integridade. Quando da não integridade desse caminho físico, a capacidade de entrada da informação será zero. Isso poderá

provocar uma desatualização do tipo de informação sendo considerado, devendo o sistema ser transferido para o estado **est9**, além de ser avisado o operador. Com relação às informações de ocupação/desocupação, posicionamento e travamento de máquinas de chave, aspecto dos sinaleiros e códigos de velocidade, no caso de não integridade do caminho físico de chegada dessas variáveis, o sistema deve colocá-las num estado de restrição, ou seja:

- **ocp**: indicar ocupação;
- **pos**: não indicar nenhum posicionamento;
- **tr**: indicar máquina de chave destravada;
- **sina, sinf**: indicar sinaleiro fechado;
- **cv**: indicar código de velocidade zero.

Com respeito às requisições de comandos, a não integridade do seu caminho físico impedirá o pedido de tal requisição. Nesse caso, deverá ser avisado o operador do sistema para as providências necessárias. O maior problema aqui refere-se aos comandos restritivos, como por exemplo, requisição de proibição de alinhamento de rota (**rpar**), que ficarão inativos por um período de tempo. Esse fato pode em si caracterizar uma situação perigosa. Nesses casos, é conveniente transferir o sistema para o estado **est8.**

Critério 3.10:

Este critério deve ser avaliado quando a capacidade máxima de entrada do sistema é excedida. Dessa forma, são analisadas as diversas entradas.

Se a capacidade máxima de entrada de qualquer tipo de informação (**ocp,pos,tr,sina,sinf,cv**,requisições por parte do operador) é excedida por algum motivo, deverá ser enviada mensagem ao operador, para se averiguarem as possíveis causas do problema, além de atitudes imediatas, para amenizar as conseqüências de tal sobrecarga. Essas atitudes dependerão muito da tecnologia e da arquitetura do sistema sendo avaliado ou especificado. No entanto, pode-se destacar algumas recomendações:

- geração de sinais de controle para sistemas externos, visando diminuir a capacidade de entrada dessa informação; ou
- requisitos para bloquear os canais em sobrecarga; ou

- requisitos para reduzir a precisão, o tempo de resposta, ou outra característica que permita ao sistema processar uma capacidade de entrada maior; ou
- requisitos para reduzir a funcionalidade do sistema; ou
- desligamento do sistema.

Em qualquer uma das atitudes descritas anteriormente, o sistema deverá ser transferido para o estado **est9**, para efeitos de segurança, já que as conseqüências da sobrecarga de entrada são, em geral, bastante problemáticas, invadindo inclusive áreas reservadas a outras informações. Vale ressaltar que a decisão final da atitude a ser tomada nesse caso deverá, em última instância, estar dependente de uma análise bastante detalhada das possíveis conseqüências de tal sobrecarga no sistema em questão.

Critério 3.11:

Quando a sistema atinge o estado **est9**, ele deve permanecer nesse estado até que as condições de segurança estejam estáveis. No caso específico de se ter atingido esse estado devido à sobrecarga na entrada do sistema, deve ser especificado um tempo de estabilidade, denominado de "histerese", a partir do qual considera-se que o fluxo de entradas já esteja dentro dos valores normais e estável. Evidentemente que a verificação dessa estabilidade é validada pelas circunstâncias que levaram o sistema para o estado **est9**, podendo depender de situações previsíveis e controláveis. No entanto, a variação na taxa de entrada das informações pode, em alguns casos, depender de condições ou de sistemas fora de controle do sistema sendo especificado ou analisado. Assim, a determinação do tempo de histerese, em função dessas considerações, deve ser realizado de acordo com as características do sistema em questão, do ambiente operacional e das condições de eventuais controles externos e que façam interface com este. Deve-se ter em mente, também, a permanência por um tempo limitado nesse estado de degradação funcional, visando não provocar outras condições inseguras.

Critério 4.1:

O que deve ser avaliado neste critério refere-se ao fato de a capacidade máxima de saída ser excedida. Nesse aspecto, convém analisar as diversas saídas deste sistema, comparando-as com a capacidade dos atuadores e as limitações do processo sendo controlado:

- Comando de posicionamento de máquina de chave (**cmmch**): considerando o tempo de deslocamento de uma máquina de chave de uma posição normal/reversa para reversa/normal ser atualmente de aproximadamente 15 segundos, deve-se evitar comandos consecutivos em intervalos de tempo menores que esse período. Caso isso aconteça, o último comando deve ser ignorado. Tal atitude caracteriza-se numa opção de ação, não se configurando numa posição padronizada, mesmo porque tal evento não caracteriza, em si, uma situação perigosa ou insegura.

- Comando de travamento/destravamento de máquina de chave (**trmch**): este comando relaciona-se com o fato de desligar/ligar o motor de travamento da máquina de chave. Assim, o limite da capacidade de saída desse comando está dependente da capacidade tolerada pelo motor de travamento sendo utilizado e, por conseqüência, do sistema de energia adotado. Vale ressaltar também que a destravamento indevido de uma máquina de chave pode vir a provocar condições perigosas e inseguras, já que a máquina de chave, estando destravada, permite a mudança de posição de sua ponta de agulha. Caso essa capacidade de saída seja excedida, o sistema deve ser transferido para o estado **est9**.

- Comando de geração de código de velocidade (**cmcv**): não há problemas específicos de capacidade de saída e, sim, aspectos operacionais relacionados com os perfis de velocidade desejados;

- Comando de Acionamento de Sinaleiros (**cmas, cmfs**): a limitação neste caso está relacionada também com os aspectos de energia do sistema, devendo ser compatível com as exigências energéticas decorrentes da abrangência do sistema. Caso essa capacidade de saída seja excedida, o sistema deve ser transferido para o estado **est9**.

Critério 4.2:

Para a verificação deste critério, será analisada cada uma das variáveis de entrada, já que as mesmas são determinantes nas gerações das diversas saídas deste sistema:

- **ocp**: na utilização de qualquer informação de ocupação/desocupação, deve existir um tempo de validade **TV** para considerar essa informação não envelhecida. O tempo de validade vai depender diretamente da velocidade comercial dos trens, do tamanho dos trechos de via e do headway do sistema. Quando do envelhecimento constatado da informação, a saída do sistema deve considerá-la na condição restritiva, ou seja,

ocupação de trecho de via, avisando o operador e transferindo o sistema para o estado **est9**;

- **pos, tr**: na utilização de qualquer informação de posicionamento e travamento de máquinas de chave, deve existir um tempo de validade **TV** para se considerar essa informação não envelhecida. O tempo de validade vai depender diretamente da velocidade comercial dos trens, do tamanho dos trechos de via e do headway do sistema. Quando do envelhecimento constatado da informação, a saída do sistema deve considerá-la na condição restritiva, ou seja, máquina de chave destravada e sem posicionamento, avisando o operador e transferindo o sistema para o estado **est9**;

- **sina, sinf**: na utilização de qualquer informação de indicação do estado do sinaleiro, deve existir um tempo de validade TV para considerar essa informação não envelhecida. O tempo de validade vai depender diretamente da velocidade comercial dos trens, do tamanho dos trechos de via e do headway do sistema. Quando do envelhecimento constatado da informação, a saída do sistema deve considerá-la na condição restritiva, ou seja, sinaleiro fechado, avisando o operador e o transferindo o sistema para o estado **est9**;

- **cv**: na utilização de qualquer informação de indicação de código de velocidade, deve existir um tempo de validade **TV** para considerar essa informação não envelhecida. O tempo de validade vai depender diretamente da velocidade comercial dos trens, do tamanho dos trechos de via e do headway do sistema. Quando do envelhecimento constatado da informação, a saída do sistema deve considerá-la na condição restritiva, ou seja, código de velocidade zero, avisando o operador e transferindo o sistema para o estado **est9**;

Com relação às requisições de comando por parte do operador, a determinação de tempos de validade não se relaciona com a segurança do sistema, ficando a critério dos usuários estabelecer esses tempos em função da operacionalidade do sistema.

Critério 4.3:

Para a verificação deste critério, é necessário determinar, em primeiro lugar, quais são as seqüências de ações perigosas. De acordo com o formalismo apresentado no capítulo anterior, essas seqüências correspondem àquelas que levam o sistema a um estado perigoso. No exemplo apresentado, não existem ações que levem o sistema a um estado perigoso, pois todos os caminhos que levam a estados desse tipo, **est8**, acontecem

devido a incongruências ou inconsistências ocorridas no sistema e, não, devido a ações determinantes.

Critério 4.4:

Para a verificação deste critério, será utilizado o mesmo método aplicado no critério 3.6, verificando os tempos de ausência ou chegada dos sinais para todos os predicados de entrada:

- **inicia↑:** não depende de outro predicado de entrada;
- **fiminicia↑:** depende do predicado de entrada **inicia↑,** já verificado no critério 3.4, estado **est1**. Como esse evento indica que o processo de iniciação já está terminado, é conveniente incluir um tempo adicional de latência **Δt**, com a finalidade de certificar-se da efetivação dos comandos realizados na iniciação;
- **termina↑:** não depende de outro predicado de entrada;
- **fimtermina↑:** depende do predicado de entrada **termina↑,** já verificado no critério 3.4, estado **est2**. Como esse evento indica que o processo de terminação já está concluído, é conveniente incluir um tempo adicional de latência **Δt**, com a finalidade de certificar da efetivação dos comandos realizados na terminação;
- **rar↑:** não depende de outro predicado de entrada;
- **alinhamentoOK↑:** depende do predicado de entrada **rar↑,** devendo ser incluído, ao tempo de duração, especificado no critério 3.6, um tempo de latência **Δt**, com o objetivo da certificação da não mudança das informações correspondentes na via, ou seja, desocupação de trechos da rota a ser alinhada e destravamento das máquinas de chave envolvidas;
- **alinhamentonãoOK↑:** não é aplicável o tempo de latência por este ser um evento restritivo, ou seja, não permitindo o alinhamento de uma rota;
- **travamentoOK↑:** depende do predicado **alinhamentoOK↑**, pois o travamento da rota só é realizado após a verificação do seu alinhamento. Deve ser acrescido, ao tempo de duração determinado no critério 3.6, um tempo de latência **Δt**, com o objetivo de certificar-se da não mudança das informações correspondentes na via, ou

seja, desocupações dos trechos da via envolvidos na rota, além do posicionamento e travamento das máquinas de chave;

- **travamentonãoOK↑:** não é aplicável o tempo de latência, por ser um evento restritivo, ou seja, não permitindo o alinhamento de uma rota;
- **ocp(1°Trecho)↑:** com relação ao alinhamento de uma rota, a não ocupação do primeiro trecho provoca uma atitude restritiva, não sendo necessário um tempo de latência;
- **ocp(últimotrecho) ↑:** como a atitude tomada na ausência deste evento é restritiva, não há a necessidade de tempo de latência;
- **rcar↑ ∧ ¬timeout↑:** não se aplica tempo de latência;
- **timeoutcanc↑:** este evento depende da ocorrência de um pedido de cancelamento, já verificado no critério 3.4, estado **est6**, sendo que a não ocorrência deste evento provoca uma atitude restritiva, não necessitando um tempo de latência.
- **rpar↑:** não depende de outro evento de entrada;
- **rcpar↑:** não depende de outro evento de entrada;
- **ocp(trechointerm)↑:** como as atitudes tomadas na ausência deste evento são restritivas, não há a necessidade de tempo de latência;
- **perigosuperado↑:** não depende de um evento de entrada específico;
- **liberação↑:** não depende de um evento de entrada específico;
- **sina↑, sinf↑ e cv↑:** o não recebimento destes eventos de entrada provoca atitudes restritivas por parte do sistema, não necessitando de tempo de latência.

Critério 4.5:

Para a determinação das ações adequadas quando ocorrem eventos dentro do tempo de latência **Δt**, é avaliado cada um dos casos que necessitam desse tempo e que foram apontados no critério anterior:

- **fiminicia↑:** se houver mudança dos estados da via durante esse tempo de latência, o sistema deve ser transferido para o estado **est9**, avisando o operador do ocorrido;

- **fimtermina↑:** se houver mudança dos estados da via durante esse tempo de latência, o sistema deve ser transferido para o estado **est9**, avisando o operador do ocorrido;

- **alinhamentoOK↑:** se houver mudanças na via durante o tempo de latência, o sistema deve permanecer no estado **est4**, avisando o operador do ocorrido e gerando o evento de saída **alinhamentonãoOK↑;**

- **travamentoOK↑:** se houver mudanças na via durante o tempo de latência, o sistema deve permanecer no estado **est4**, avisando o operador do ocorrido e gerando o evento de saída **travamentonãoOK↑**.

Critério 4.6:

Com relação ao sistema apresentado, existe uma decisão crítica na interface homem-máquina:

- Requisição para Cancelamento de Proibição de Alinhamento de Rota **rcpar**, seguida por uma Requisição para Proibição de Alinhamento de Rota **rpar**: vale ressaltar que esse caso caracteriza-se por um arrependimento na ação de cancelar uma proibição; na realidade, essa proibição, por algum motivo, não deveria ser cancelada. A título de ilustração, pode ser apresentada a colocação de uma proibição de alinhamento de rota numa região em que existe pessoal de manutenção na via. Certo momento, o operador retira essa proibição quando, de fato, ainda existe pessoal de manutenção ainda na via. Assim, o operador tenta rapidamente corrigir o seu erro recolocando a proibição. Nesses casos críticos, deve ser estabelecido um fator de latência Δt, durante o qual o recebimento de um novo comando de arrependimento evita o efeito do comando anterior, erroneamente requisitado. Assim, nesse exemplo, deve ser estabelecido um tempo de arrependimento, ou de latência, para a efetivação da requisição de cancelamento de proibição de alinhamento de rota. Se durante esse tempo de latência chegar uma outra requisição para proibição de alinhamento de rota, a requisção de cancelamento de proibição não deverá ser efetivada. Esse tempo deve ser determinado em função dos aspectos operacionais, como valor do headway, perfil dos passageiros, entre outros.

Critério 4.7:

Para cada requisição do operador que se efetua em função da ocorrência do recebimento de informações através de mensagens e/ou avisos na interface homem/máquina, deve ser especificado um período de histerese correspondente ao atraso da ação do operador na interpretação dessas informações. Esse aspecto deve ser lembrado principalmente quando a segurança do sistema depender de ações do operador deduzidas a partir das informações por ele recebidas. Nesse sentido, quanto maior for a facilidade de interpretação das informações, menor será esse período de histerese. Quando do estudo da segurança de sistemas, o tempo de histerese deverá, obrigatoriamente, ser considerado na temporização dos eventos, podendo inclusive levar a condições perigosas caso o mesmo seja muito longo em relação às exigências do ambiente de aplicação. Em sistemas metro-ferroviários esse tempo máximo de histerese dependerá muito da velocidade dos trens, do headway da linha, entre outros fatores. Para efeito de ilustração, um trem a velocidade de 80Km/h percorrerá 111 metros em 5 segundos. Se o trecho de via correspondente a uma ocupação for da ordem de 250 metros, evidentemente o período máximo de histerese para decisões críticas não deve passar de 5 segundos, assegurando, dessa forma, a não ocupação e a posterior desocupação de um trecho de via inteiro dentro desse intervalo. Esse exemplo tem apenas um objetivo: o de ilustrar a questão, sem querer, com isso, estabelecer valores determinantes, já que, na prática, a determinação adequada desses limites envolverá uma grande quantidade de parâmetros operacionais e parâmetros decorrentes da arquitetura e da tecnologia adotada.

Critério 5.1:

Para a verificação deste critério, deve ser avaliada a existência de variáveis de supervisão correspondentes às variáveis de controle. Dessa forma, para cada variável de saída do controlador, será verificada a existência ou não da correspondente variável de entrada de supervisão:

- Comando para máquina de chave normal/reverso (**cmmch**): a variável de supervisão correspondente é **pos**, que indica a posição da respectiva máquina de chave;
- Comando de travamento/destravamento para máquina de chave (**trmch**): a variável de supervisão **tr** corresponde à indicação do travamento ou do destravamento da máquina de chave;

- Comando de Geração de Código de Velocidade (**cmcv**): a variável de supervisão **cv** corresponde ao código de velocidade efetivamente colocado num determinado trecho da via;

- Comando de Acionamento de Abertura de Sinaleiro (**cmas**): a variável de supervisão **sina** indica que o sinaleiro está aberto;

- Comando de Acionamento de Fechamento de Sinaleiro (**cmfs**): a variável de supervisão **sinf** indica que o sinaleiro está fechado.

Critério 5.2:

Para a verificação deste critério, será avaliado o período entre a supervisão de um evento, efetuada por uma variável de supervisão, correspondente a uma ação solicitada por uma variável de controle:

- A variável de supervisão **pos** está associada à variável de controle **cmmch**. Deve ser estabelecido um tempo mínimo e um tempo máximo para recebimento dessa indicação. O tempo mínimo está relacionado com o atraso mínimo do sistema e também com o tempo mínimo de deslocamento da ponta de agulha da máquina de chave. Por outro lado, o tempo máximo está relacionado com o atraso máximo do sistema e o tempo máximo de deslocamento da ponta de agulha. Evidentemente, os tempos envolvidos com o deslocamento da ponta de agulha serão determinantes, por apresentarem um valor absoluto maior do que o atraso do sistema de controle. Uma resposta prematura poderá indicar problemas no sistema de supervisão, sendo adequada a transferência do sistema para o estado **est9**. Por sua vez, uma resposta demorada nessa informação implica gerar sinais de controle do tipo **alinhamentonãoOK↑, travamentonãoOK↑** e não gerar sinais de controle do tipo **perigosuperado↑** ou **liberação↑,** além da anulação do respectivo comando. Além disso, no caso de resposta demorada, é importante também enviar mensagem ao operador.

- A variável de supervisão **tr** está associada à variável de controle **trmch**. Da mesma forma, deve ser estabelecido um tempo mínimo e um tempo máximo para a supervisão dessa informação. O tempo mínimo e o tempo máximo estão relacionados com os tempos de atraso no sistema e com os tempos para travamento e destravamento da máquina de chave, que estão diretamente relacionados com as características do motor de acionamento da mesma. Se uma informação de supervisão

de travamento/destravamento for recebida precocemente, isso pode indicar problemas no sistema de supervisão e, portanto, o sistema deve ser transferido para o estado **est8**, por caracterizar uma situação insegura. Caso essa informação seja recebida de forma atrasada, o sistema deve gerar informações do tipo **alinhamentonãoOK↑**, **travamentonãoOK↑** e não gerar informações do tipo **perigosuperado↑** e **liberação↑**. Além disso, no caso de resposta demorada, é importante também informar o operador do ocorrido.

- As variáveis **sina** e **sinf** indicam estados dos sinaleiros e são variáveis de supervisão em relação ao comando **cmas** e **cmfs**. Da mesma forma que nos casos anteriores, deve ser estabelecido um tempo mínimo e um tempo máximo de resposta ao comando. Esses tempos estão relacionados com os atrasos do sistema e com os atrasos da abertura e do fechamento dos sinaleiros. Um recebimento precoce dessa informação implica problemas no sistema de supervisão, devendo o sistema ser transferido para o estado **est8**, por caracterizar uma situação perigosa. Um recebimento tardio dessa informação indica também problemas na supervisão, sendo que o sistema deve ser transferido para o estado **est9**.

- A informação de Indicação de Código de Velocidade **cv** representa a variável de supervisão do comando de geração de código de velocidade **cmcv**. Deve também ser estabelecido, nesse caso, um tempo mínimo e máximo, correspondentes aos atrasos do sistema e aos atrasos dos transmissores e receptores de código de velocidade. Tanto uma resposta precoce como uma resposta demorada indicam problemas na supervisão da velocidade no trecho de via correspondente. Tal situação requer que o sistema seja transferido para o estado **est9**.

Critério 5.3:

Para a verificação deste requisito, será avaliada a ocorrência de um evento de entrada não esperado, ou seja, a ocorrência de mudança em alguma variável supervisionada, sem o correspondente comando:

- Mudanças ocorridas na variável de supervisão **pos**, sem um correspondente comando de máquina de chave **cmmch**, indicam a existência de possíveis interferências no sistema. Nesse caso, devem ser enviadas mensagens ao operador e o sistema deve ser transferido para o estado **est8**, por caracterizar uma situação perigosa.

- Mudanças ocorridas na variável de supervisão **tr**, sem um correspondente comando de máquina de chave **trmch**, indicam a existência de possíveis interferências no sistema. Nesse caso, devem ser enviadas mensagens ao operador e o sistema deve ser transferido para o estado **est8,** por caracterizar situação perigosa.

- Mudanças ocorridas na variável de supervisão **sina** ou **sinf**, sem um correspondente comando de acionamento de sinaleiro **cmas** ou **cmfs**, indicam a existência de possíveis interferências no sistema. Nesse caso, devem ser enviadas mensagens ao operador e o sistema deve ser transferido para o estado **est8,** por caracterizar situação perigosa.

- Mudanças ocorridas na variável de supervisão **cv**, sem um correspondente comando de geração de código de velocidade **cmcv**, indicam a existência de possíveis interferências no sistema. Nesse caso, devem ser enviadas mensagens ao operador e o sistema deve ser transferido para o estado **est8**, por caracterizar situação perigosa.

- Mudanças ocorridas na variável de ocupação **ocp,** sem haver uma rota alinhada aprovada, indicam a existência de incongruências no sistema. Nesse caso, devem ser enviadas mensagens ao operador e o sistema deve ser transferido para o estado **est8,** por caracterizar situação perigosa.

Critério 6.1:

Para avaliar este critério, será analisado, para cada estado, os caminhos, a partir do **est0**, para serem atingidos esses estados. A identificação desse caminho é formalizada através da seguinte padronização:

<estj>/<predicado de entrada i>/<estk>/<predicado de entrada l>/...../<estm>

- **est0**: Caminho apresentado

est0/termina↑/est0;

- **est1**: Caminho apresentado

est0/inicia↑/est1;

- **est2**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2;

- **est3**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/termina↑/est3;

- **est4**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/rar↑/est4;

- **est5**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/ocp(1°Trecho)↑/est5;

- **est6**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/(rcar↑ ∧ timeout↑)/est6;

- **est7**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/rpar↑/est7;

- **est8**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/ocp(trechointerm)↑/est8;

- **est9**: Caminho apresentado

est0/inicia↑/est1/fiminicia↑/est2/ocp(trechointerm)↑/est8/perigosuperado↑/est9.

Critério 6.2:

Para avaliar este critério, é analisada, para cada um dos estados, a existência de um caminho, não vazio, que leve o sistema para ele mesmo. Dessa forma, identificar-se-á um ciclo nas tabelas de transição de estados e a sua coerência, para, assim, comprovar este critério. A identificação desse ciclo é formalizada através da seguinte padronização:

<estj>/<predicado de entrada i>/<estk>/<predicado de entrada I>/...../<estj>

- **est0**: Ciclo identificado

est0/inicia↑/est1/fiminicia↑/est2/termina↑/est3/fimtermina↑/est0;

coerência(inicia↑ ∧ fiminicia↑ ∧ termina↑ ∧ fimtermina↑) = T;

- **est1**: Ciclo identificado

est1/fiminicia↑/est2/termina↑/est3/fimtermina↑/est0/inicia↑/est1;

coerência(fiminicia↑ ∧ termina↑ ∧ fimtermina↑ ∧ inicia↑) = T;

- **est2**: Ciclo identificado

est2/ocp(1°Trecho)↑/est5/ocp(últimotrecho)↑/est2;

coerência(ocp(1°Trecho)↑ ∧ ocp(últimotrecho)↑) = T;

- **est3**: Ciclo identificado

est3/fimtermina↑/est0/inicia↑/est1/fiminica↑/est2/termina↑/est3;

coerência(fimtermina↑ ∧ inicia↑ ∧ fiminicia↑ ∧ termina↑) = T;

- **est4**: Ciclo identificado

est4/travamentoOK↑/est2/rar↑/est4;

coerência(travamentoOK↑ ∧ rar↑) = T;

- **est5**: Ciclo identificado

est5/ocp(últimotrecho)↑/est2/ocp(1°Trecho)↑/est5;

coerência(ocp(últimotrecho)↑ ∧ ocp(1°Trecho)↑) = T;

- **est6**: Ciclo identificado

est6/timeoutcanc↑/est2/rcar↑ ∧ timeout↑/est6;

coerência(timeoutcanc↑ ∧ rcar↑ ∧ timeout↑) = T;

- **est7**: Ciclo identificado

est7/rcpar↑/est2/rpar↑/est7;

coerência(rcpar↑ ∧ rpar↑) = T;

- **est8** : não existe caminho coerente, conforme explicado na verificação do critério 6.4;
- **est9**: não existe caminho coerente, pois este só é atingido em condições anormais do sistema, conforme descrito no detalhamento de cada estado.

Critério 6.3:

Para a avaliação deste critério, deve ser analisada cada uma das requisições de comando presentes no sistema e a possível existência da requisição reversa:

- **inicia** e **termina**: estes comandos atendem este critério, de forma que existem caminhos entre os estados atingidos e revertidos;

est0/inicia↑/est1/fiminicia↑/est2 (iniciação);

coerência(inicia↑ ∧ fiminicia↑) = T;

est2/termina↑/est3/fimtermina↑/est0 (terminação);

coerência(termina↑ ∧ fimtermina↑) = T;

- Requisição para Alinhamento de Rota (**rar**) e Requisição para Cancelamento de Alinhamento de Rota (**rcar**): estes comandos atendem este critério, de forma que existem caminhos entre os estados atingidos e revertidos;

est2/rar↑/est4/alinhamentoOK↑/est4 (alinhamento aprovado);

coerência(rar↑ ∧ alinhamentoOK↑) = T;

est4/rcar↑ ∧ timeout↑/est2 (cancelamento de alinhamento);

coerência(rcar↑ ∧ timeout↑) = T;

- Requisição para Proibição de Alinhamento de Rota (**rpar**) e Requisição para Cancelamento de Proibição de Alinhamento de Rota (**rcpar**): estes comandos atendem este critério, de forma que existem caminhos entre os estados atingidos e revertidos;

  est2/rpar↑/est7 (proibição de alinhamento de rota);

  coerência(rpar↑) = T;

est7/rcpar↑/est2 (cancelamento de proibição de alinhamento de rota;

coerência(rcpar↑) = T;

Critério 6.4:

Para avaliar este critério, deve-se verificar a não existência de um caminho ou a não coerência de uma seqüência de predicados de entrada que levam o sistema para um estado perigoso ou para um estado inseguro, características detectadas pelo estado **est8**. O fato de se verificar a não coerência é devido à existência de caminhos possíveis para esse estado **est8**:

- **est1/ocp(1°Trecho)↑/est8**: a ocorrência do predicado de entrada **ocp(1°Trecho)↑** não é coerente se o sistema estiver no estado **est1** de iniciação, já que o sistema está em processo de iniciação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est1/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est1** de iniciação, já que o sistema está em processo de iniciação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est1/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est1** de iniciação, já que o sistema está em processo de iniciação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est2/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est2** de espera, já que aguarda-se ou a invasão de uma rota pelo trem ou alguma requisição do operador. Esse predicado deve ter acontecido por algum evento de falha no sistema, provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est2/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est2** de espera, já que aguarda-se ou a invasão de uma rota pelo trem ou alguma requisição do operador. Esse predicado deve ter acontecido por algum evento de falha no sistema, provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est3/ocp(1°Trecho)↑/est8**: a ocorrência do predicado de entrada **ocp(1°Trecho)↑** não é coerente se o sistema estiver no estado **est3** de terminação, já que o sistema está em processo de terminação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est3/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est3** de terminação, já que o sistema está em processo de terminação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est3/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est3** de terminação, já que o sistema está em processo de terminação e não devem estar acontecendo eventos que serão controlados por ele, como por exemplo, a movimentação de trens;

- **est4/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est4** de alinhamento, já que o sistema está em processo de aprovação de uma rota. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est4/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est4** de alinhamento, já que o sistema está em processo de aprovação de uma rota. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

  est4/ocp(1°Trecho)↑/est8: a ocorrência do predicado de entrada **ocp(1°Trecho)↑** não é coerente se o sistema estiver no estado **est4** de alinhamento, já que o sistema está em processo de aprovação de uma rota. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est5/travamentonãoOK↑/est8**: a ocorrência do predicado de entrada **travamentonãoOK↑** não é coerente se o sistema estiver no estado **est5** de cancelamento automático, já que, nesse caso, o trem já invadiu uma rota alinhada e

travada. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est6/travamentonãoOK↑/est8**: a ocorrência do predicado de entrada **travamentonãoOK↑** não é coerente se o sistema estiver no estado **est6** de cancelamento, já que, nesse caso, foi pedido o cancelamento de uma rota em que já foram aprovados o seu alinhamento e travamento. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est6/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est6** de cancelamento, já que, nesse caso, aguarda-se a não invasão de um trem pela rota ou, no máximo, a ocupação do primeiro trecho de via, caso não haja tempo para o trem respeitar o pedido de cancelamento. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est6/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est6** de cancelamento, já que aguarda-se a não invasão de um trem pela rota ou, no máximo, a ocupação do primeiro trecho de via, caso não haja tempo para o trem respeitar o pedido de cancelamento. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externoos;

- **est7/ocp(1°Trecho)↑/est8**: a ocorrência do predicado de entrada **ocp(ocp1°Trecho)↑** não é coerente se o sistema estiver no estado **est7** de proibição de alinhamento, já que, nesse caso, não deve ser alinhada a rota, muito menos invadida por um trem. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externoos;

- **est7/ocp(últimotrecho)↑/est8**: a ocorrência do predicado de entrada **ocp(últimotrecho)↑** não é coerente se o sistema estiver no estado **est7** de proibição de alinhamento, já que, nesse caso, não deve ser alinhada a rota, muito menos invadida por um trem. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

- **est7/ocp(trechointerm)↑/est8**: a ocorrência do predicado de entrada **ocp(trechointerm)↑** não é coerente se o sistema estiver no estado **est7** de proibição de alinhamento, já que, nesse caso, não deve ser alinhada a rota, muito menos invadida por um trem. Esse predicado deve ter acontecido por algum evento de falha no sistema provocado por problemas internos ao mesmo ou por distúrbios externos;

Critério 6.5 e Critério 6.6:

No caso do sistema em questão, o estado perigoso corresponde ao estado **est8**. O estado seguro, só que degradado, corresponde ao estado **est9**. O sistema é transferido do estado **est8** para o estado **est9** quando a situação de perigo envolvendo o sistema tiver sido superada, o que corresponde à ocorrência do evento **perigosuperado↑**. No entanto, enquanto o perigo não for superado, pode ocorrer algum outro evento, interno ou externo ao sistema, e conduzi-lo para um estado inseguro. Todas essas possibilidades estão relacionadas com a probabilidade do sistema em atingir um estado inseguro. Essa probabilidade deve estar relacionada com o mínimo risco aceitável para o sistema, que é determinado fundamentalmente pelo valor do MTBUF, calculado através do modelamento global do sistema.

Em função de todas as avaliações aqui realizadas, conclui-se pela perfeita adaptabilidade e usabilidade dos critérios, abrindo também uma grande perspectiva de automação do processo de verificação, auxiliado por ferramentas de simulação associadas a ferramentas **CASE**s, já existentes e que auxiliam no desenvolvimento de projetos.

Vale ressaltar, também, que a avaliação integral dos critérios é possível desde que realizada nos diversos níveis de especificação, principalmente no que diz respeito ao software, em função da complexidade e não padronização de seu controle. No entanto, mesmo com a aplicação dos critérios num modelo adotado de sistema relativamente simples, obteve-se um retorno do processo de análise bastante eficaz. Todos esses resultados comprovam que a metodologia de avaliação da segurança de sistemas críticos, proposta nesta tese, pode efetivamente contribuir na melhoria da segurança desses sistemas, tornando a avaliacão da segurança um método mais sistemático, mais objetivo e mais completo.

6. CONCLUSÕES

Uma das contribuições deste trabalho refere-se ao desenvolvimento de uma metodologia de avaliação da segurança baseada em novos conceitos, bem como de sua plena aplicação num sistema modelo de controle metro-ferroviário, procurando destacar os principais cuidados a serem tomados, tanto no desenvolvimento como na análise desses sistemas.

Como apresentado no início deste trabalho, o principal objetivo foi a criação de uma metodologia de avaliação da segurança, com enfoque principal na completeza das especificações de sistemas críticos. Dentro dessa nova filosofia de trabalho, são apresentadas, inicialmente, neste capítulo, as contribuições e conclusões a respeito desta metodologia, destacando os seguintes aspectos: novo enfoque, padronização utilizada e critérios de verificação e sua aplicabilidade.

Em seguida, são discutidos os benefícios obtidos com os resultados da aplicação desta metodologia em Sistemas de Controle Metro-Ferroviários.

Finalizando, são apontadas as linhas de pesquisa futuras e os trabalhos que podem ser realizados em continuidade a esta tese de doutorado.

A metodologia desenvolvida neste trabalho teve como principal enfoque englobar os aspectos que ainda provocam falhas perigosas e inseguras em sistemas de controle críticos. Em função de diversos estudos realizados e apontados ao longo deste trabalho, concluiu-se que a falta de completeza da especificação desses sistemas é uma das principais causas desses tipos de falhas. Esta metodologia desenvolveu, assim, critérios de completeza não apenas sobre programas de software críticos, mas fundamentalmente sobre sistemas críticos, enquadrando-a de forma coerente dentro de um Programa Global de Segurança.

Nesse sentido, esta metodologia adotou como ponto de partida os diagramas de fluxo de dados estendidos do sistema, podendo e devendo ser aplicada a todos os níveis de abstração desses diagramas, inclusive àqueles imediatamente anteriores à etapa de implementação, seja ela de software ou de hardware. Evidentemente que, a partir da implementação, os métodos de análise diferem, conforme apontado no capítulo 3 desse trabalho.

Dentro desta visão metodológica, pode-se também contar com a participação do engenheiro da aplicação, componente fundamental quando se pretende avaliar a completeza de uma especificação dentro de uma visão sistêmica.

Para se implementar esta metodologia, foi necessária a criação de uma padronização. Essa padronização apresenta características informais em conjunto com características formais.

Conforme pesquisas realizadas e apontadas ao longo desta tese, os aspectos informais preenchem as lacunas ainda não plenamente e adequadamente contempladas pelos métodos formais. Dentro desse enfoque, enquadram-se uma melhor compreensão da funcionalidade do sistema, bem como a criação de critérios que visam verificar a completeza das especificações de sistemas críticos. Assim sendo, os diagramas de fluxo de dados estendidos se aplicam de maneira adequada na descrição da funcionalidade do sistema, tendo como produto o modelo de transição de estados, que pode ser implementado através das tabelas de transição de estados, sobre as quais os critérios são aplicados. Em função dos resultados práticos obtidos com a aplicação dos critérios adotados, pode-se também auxiliar na melhoria do grau de abrangência dos mesmos.

A padronização formal desta metodologia teve como objetivo primordial permitir a clara inteligibilidade dos critérios e sua aplicabilidade sobre o modelo de transição de estados de um sistema. Esse modelo pode também e deve servir de base para a avaliação global da segurança de um sistema crítico, englobando as atividades de análise de segurança sobre uma única base de referência. Em função das necessidades encontradas, foi adotada, como linguagem formal para essa padronização, as expressões de lógica de predicados, denominadas WFF, acrescidas de funções típicas do modelo de transição de estados.

Ainda com relação às contribuições, vale destacar o desenvolvimento de critérios de completeza, que podem ser aplicados a diversos sistemas críticos, bem como em outros sistemas em que aspectos como confiabilidade e disponibilidade são importantes. Nesses sistemas que não envolvem o atributo segurança, a aplicação desta metodologia deve ser avaliada também em função dos custos envolvidos com sua aplicação. Nos sistemas de segurança, esses critérios devem ser aplicados em todos os níveis de especificação, visando um melhor resultado no processo de avaliação da segurança. Essa aplicabilidade em nível de sistema, não encontrada de maneira clara na literatura técnica, ficou

plenamente comprovada através dos resultados obtidos por sua aplicação num sistema modelo de controle metro-ferroviário apresentado no capítulo 5 desta tese.

Outra contribuição importante deste trabalho refere-se à utilização desses critérios também na determinação de novos Requisitos Gerais de Segurança (**RGS**) relacionados com o sistema de controle e supervisão. A experiência tem demonstrado que as técnicas que auxiliam a determinação dos **RGS**, entre elas a árvore de falhas, auxiliam, de forma predominante, na determinação dos **RGS** mais relacionados com o processo sendo controlado e menos com o sistema de controle e supervisão propriamente dito. Essa conclusão ficou bastante evidenciada em função dos resultados obtidos com a aplicação dos critérios num sistema modelo de controle metro-ferroviário, realizada neste trabalho.

A metodologia de avaliação da segurança proposta nesta tese pode ser aplicada tanto em sistemas sendo projetados como em sistemas já implementados e que devam ser analisados do ponto de vista de segurança. Assim sendo, os resultados obtidos da aplicação desses critérios num sistema modelo de controle metro-ferroviário podem ser utilizados como referência para a elaboração de um "Guia de Segurança" no desenvolvimento de novos sistemas de controle metro-ferroviários, evidentemente complementado por um conjunto de Requisitos Gerias de Segurança decorrentes do processo sendo controlado, como apresentado no Anexo B deste trabalho. Esse "Guia de Segurança" deve fazer parte de um Programa Global de Segurança, conforme evidenciado no capítulo 3 desta tese.

Com relação às pesquisas futuras que podem ser realizadas para continuidade das pesquisas efetuadas, vale ressaltar a tarefa de verificação de completeza dos próprios critérios aqui desenvolvidos. Essa verificação deve ser realizada sob dois enfoques não excludentes. O primeiro está relacionado com a verificação formal da completeza dos critérios sob o modelo de transição de estados, tendo como base a análise dos caminhos possíveis e independentes. Outro enfoque deve ser decorrente da aplicação prática desses critérios em sistemas críticos, verificando, dessa forma, possíveis lacunas ainda não contempladas por esses critérios, mas que são importantes para uma verificação adequada da segurança dos sistemas. Nesse sentido, após uma comprovada adequação prática do modelo informal ao ambiente de aplicação, pode-se, então, iniciar o processo de formalização do ambiente em questão e, por conseqüência, da avaliação da completeza dos próprios critérios.

Outra atividade de continuidade deste trabalho refere-se à implementação de uma ferramenta que automatize a aplicação dos critérios propostos sob um modelo de transição de estados. Nesse sentido, foi apresentado, no Anexo A deste trabalho, o princípio básico de conversão de expressões lógicas em regras implementadas em linguagem Prolog. Esse método de trabalho pode servir de base para o desenvolvimento de um protótipo que trabalhe em conjunto com ferramentas CASEs já existentes e que automatizem, de alguma forma, os métodos de diagrama de fluxo de dados e modelo de transição de estados.

Enfim, pode-se afirmar que os objetivos desta pesquisa foram plenamente alcançados em função dos resultados obtidos de sua aplicação e da perspectiva de utilização na avaliação de diversos tipos de sistemas críticos. Inicialmente, existiam dúvidas relacionadas com a perfeita adequação desta metodologia em sistemas críticos, principalmente em nível de especificação de sistemas. Essas questões eram suportadas pela falta de informações detalhadas numa especificação de nível mais elevado. No entanto, verificou-se que, mesmo a partir de uma especificação relativamente simples, a verificação dos critérios provocou uma análise detalhada, não facilmente atingível somente através de outros métodos abordados neste trabalho.

Acredita-se que este trabalho colabore efetivamente na melhoria dos métodos de avaliação da segurança de sistemas críticos, especialmente dos sistemas de controle metro-ferroviários, cada vez mais importantes e necessários nessa nossa sociedade moderna.

ANEXO A

PROCESSO DE TRANSFORMAÇÃO DE EXPRESSÕES WFF EM REGRAS PROLOG

1. INTRODUÇÃO

Os predicados de entrada e os predicados de saída são representados através de expressões de lógica de predicados. A lógica de predicados é um dos métodos mais simples para a representação da declaração de um determinado conhecimento.

A lógica de predicados fornece símbolos para representar objetos e, então, permite que esses objetos sejam utilizados como componentes de uma declaração. Como exemplo, pode-se ter uma constante **a** e um predicado **P**, para representar a declaração **P(a).**

Para facilitar as explicações deste apêndice, são adotadas algumas padronizações. Os símbolos **a, b, c,** ... são utilizados na lógica de predicados para denotar objetos particulares em algum domínio. Os símbolos **P, Q, R,** ... são usados para denotar qualidades ou atributos de objetos ou relações entre objetos que são verdadeiras ou falsas. Por exemplo, **Q(x,y)** indica alguma relação entre **x** e **y**. Os símbolos de função **f, g, h,** ... denotam o mapeamento de elementos de um domínio em elementos de outro domínio, que pode ser o mesmo do primeiro. Dessa forma, **P(a,f(b))** indica que **P** é verdadeiro sobre um par de argumentos composto por **a**, seguido do valor da função **f** aplicada sobre **b**.

Os conectivos lógicos são os mesmos da lógica : $\wedge$, $\vee$, $\rightarrow$, $\neg$

Os símbolos que representam variáveis x, y, z, x1, x2, ... representam potencialmente qualquer elemento do domínio e permitem a formulação de declarações genéricas sobre alguns elementos do domínio num determinado momento.

Dois quantificadores ,$\forall$ **e** $\exists$ podem ser utilizados para construir novas declarações.

Dessa forma, a declaração $\exists$ **x P(x)** expressa que existe pelo menos um elemento do domínio que torna a declaração **P(x)** verdadeira.

A declaração $\forall$ **x P(x)** denota que qualquer elemento do domínio torna a declaração **P(x)** verdadeira.

A lógica de predicados pode ser uma representação conveniente para fatos e regras de inferência desde que exista um conjunto adequado de funções e predicados que torne possível a construção de fórmulas ou declarações.

As regras básicas para se construir fórmulas sintaticamente corretas são:[RICH 83]

1. qualquer constante ou variável é denominada um **termo**;

2. uma **função** aplicada sobre **termos** é um **termo**;

3. um **predicado** aplicado sobre **n termos** é uma **"well-formed formula"**, denominada também **wff**;

4. qualquer combinação de **wff** é também uma **wff**, sendo que todos os conectivos lógicos podem ser utilizados;

5. qualquer **wff** denominada **F** pode ser transformada em uma outra **wff** através da prefixação com um quantificador e uma variável individual, ou seja, $\forall$ **x (F)** ou $\exists$ **x (F).** Os parênteses devem ser utilizados quando necessários, para tornar claro o escopo do quantificador utilizado.

2. CONVERSÃO DE UMA WFF PARA A FORMA DE CLÁUSULA

Para utilizar uma **wff** num processo de resolução, ou seja, de verificação, é necessário simplificar sua estrutura. Essa nova estrutura será mais simples de ser trabalhada se:

- existisse um menor número de componentes;

- os quantificadores estivessem separados do resto da fórmula de maneira que não precisassem ser considerados.

A forma normal conjuntiva, criada por Davis em 1960, atende a essas propriedades. Para converter uma **wff** nessa forma normal conjuntiva, devem ser cumpridas as seguintes etapas:[RICH 83]

1. Eliminar o símbolo$\Rightarrow$: $\mathbf{a \Rightarrow b \rightarrow \neg a \vee b.}$

2. Reduzir o escopo da símbolo ¬:

   ¬(¬p) = p

   ¬(a ∧ b) = ¬a ∨ ¬b (lei de Morgan)

   ¬(a ∨ b) = ¬a ∧ ¬b (lei de Morgan)

   ¬∀ x P(x) = ∃ x ¬P(x)

   ¬∃ x P(x) = ∀ x ¬P(x)

3. Padronizar as variáveis de maneira que cada quantificador esteja ligado a uma única variável:

   ∀ x P(x) ∨ ∀ x Q(x) → ∀ x P(x) ∨ ∀ y Q(y)

   As variáveis ligadas aos quantificadores são hipotéticas, ou seja, "dummy" variáveis.

4. Colocar todos os quantificadores à esquerda da fórmula sem mudar sua ordem. Isso é possível desde que não haja conflito entre os nomes das variáveis. Por isso a necessidade do passo anterior:

   ∀ x P(x) ∨ ∀ y Q(y) → ∀ x ∀ y (P(x) ∨ Q(y))

5. Eliminar os quantificadores existenciais. Pode-se realizar tal transformação substituindo a variável por uma função que produza o valor desejado:

   ∀ x ∃ y pai(y,x) → ∀ x pai(g(x),x)

6. Eliminar todos os quantificadores universais ∀:

   ∀ x pai(g(x),x) → pai(g(x),x)

7. Converter a fórmula num conjunto de disjunções:

   (... ∨ ...∨...) ∧ (... ∨ ... ∨ ...)

   (a ∧ b) ∨ c ⇒ (a ∨ c) ∧ (b ∨ c)

8. Denominar cada elemento desse conjunto de uma cláusula separada. Para a expressão **wff** ser verdadeira, todas as cláusulas geradas devem ser verdadeiras.

Após aplicar todos esses passos, tem-se as denominadas forma normal conjuntiva ou **forma de cláusulas**.

3.CONVERSÃO DE UMA FORMA DE CLÁUSULA EM REGRAS PROLOG

Existem algumas regras básicas que podem auxiliar na conversão das formas de cláusulas em regras implementadas na linguagem Prolog. Para efeito de padronização, a avaliação de uma forma de cláusula pode ser enquadrada dentro do seguinte padrão:

avalia_forma :- <forma de cláusula convertida em prolog>

Essas regras básicas de conversão são:

- **P** $\wedge$ **Q** deve ser convertido para a regra:

  avalia_forma :- P,Q.

- **P** $\vee$ **Q** deve ser convertido para a regra:

  avalia_forma :- P;Q.

- **number_of(X,P,N)** deve ser convertido para a regra:

  avalia-forma :- set_of(X,P,S), size(S,N).

  Obs: **number-of(X,P,N)** é lido como "o número dos **X**'s que satisfazem **P** é **N**". As funções **set_of** e **size** fazem parte da biblioteca de funções da linguagem Prolog.

ANEXO B

REQUISITOS GERAIS DE SEGURANÇA PARA UM SISTEMA DE CONTROLE METRO-FERROVIÁRIO

1. INTRODUÇÃO

Este anexo descreve os Requisitos Gerais de Segurança (**RGS**) aplicáveis a um Sistema de Controle Metro-Ferroviário. O objetivo da apresentação desses requisitos é ilustrar este trabalho com um conjunto de **RGS** aplicáveis a diversos sistemas de sinalização metro-ferroviária. Nesse sentido, tais requisitos não constituem em um padrão adotado pelas empresas operadoras, mas definem um conjunto coerente de exigências de segurança.

2. CONCEITOS GERAIS DE SEGURANÇA

Do ponto de vista de segurança, os requisitos apresentados neste apêndice tiveram como referência três conceitos gerais de segurança, que se relacionam com o sistema de sinalização:

- evitar colisões;
- evitar descarrilhamentos; e
- evitar atropelamentos.

Nesse sentido, o sistema deve executar, de forma segura, as funções de controle de movimentação dos trens através das funções de :

- Proteção dos Aparelhos de Mudanças de Via, nos quais estão localizadas as Máquinas de Chave;
- Controle de Tráfego; e
- Seleção dos Códigos de Velocidade.

Vale destacar aqui que, num processo de avaliação da segurança de todos os subsistemas envolvidos no controle metro-ferroviário, é fundamental incluir na análise de riscos a condição de **fogo**, conforme considerado na análise do sistema de controle do Eurotúnel [THE CHANNEL TUNEL 94].

3. DESCRIÇÃO DOS REQUISITOS GERAIS DE SEGURANÇA

Este capítulo apresenta alguns Requisitos Gerais de Segurança (**RGS**), mostrando, em cada caso, a situação em que se aplica no sistema [FDTE 542.401.AS.001].

RGS1: Só pode haver alinhamento de uma rota em condições normais do sistema se não houver tráfego estabelecido no sentido oposto ao bloqueio de saída dessa rota.

RGS2: Se houver proibição de um bloqueio como saída de uma rota, não poderá haver alinhamento da rota que utilize esse bloqueio como saída.

RGS3: Se houver proibição de um bloqueio como entrada de uma rota, não poderá haver alinhamento de rota que utilize esse bloqueio como entrada.

RGS4: Só poderá haver alinhamento de rota em condições normais do sistema se todos os circuitos de via pertencentes à rota estiverem desocupados.

RGS5: Só poderá haver alinhamento de rota se não houver outra rota conflitante com a primeira. Define-se como Rota Conflitante qualquer rota que possa vir a provocar uma colisão em relação a uma outra rota.

RGS6: Um bloqueio só pode ser aberto se todas as máquinas de chave envolvidas estiverem eletricamente travadas em posições que definam uma rota prevista no intertravamento.

RGS7: Só poderá haver cancelamento de uma rota alinhada por desocupação seqüêncial, ou seja, cancelamento automático, ou por cancelamento pelo operador.

RGS8: O cancelamento de rota pelo operador não deverá ser efetivado se os circuitos de via pertencentes à rota alinhada já tiverem sido ocupados pelo trem.

RGS9: O cancelamento de uma rota pelo operador só deverá ser efetivado após uma temporização suficiente para parar o trem antes que ele ultrapasse o bloqueio de entrada. Essa temporização deverá ser de 60 segundos. Se o trem não estiver a uma distância do

bloqueio que permita a sua parada, o trem irá ocupar os circuitos de via pertencentes à rota e o RGS8 deverá ser garantido.

RGS10: No cancelamento de uma rota por desocupação seqüêncial, cada circutio de via só deverá deixar de fazer parte da rota após ter sido desocupado pelo trem.

RGS11: Só poderá haver destravamento de uma máquina de chave se esta não pertencer a nenhuma rota e o circuito de via da máquina de chave estiver desocupado.

RGS12: O perfil seguro de velocidade que se antecede a uma ocupação deve ser respeitado pelo intertravamento.

RGS13: O perfil seguro de velocidade que antecede a um bloqueio fechado deve ser respeitado pelo intertravamento.

RGS14: No alinhamento de rota, a seleção de código de velocidade deverá obedecer ao sentido de tráfego estabelecido e ao perfil de velocidade imposto pelas condições da via.

RGS15: O trecho correspondente ao fim de via deverá estar com código de velocidade de 0 Km/h e com sinalização apropriada.

RGS16: Só poderá haver geração de código de velocidade superior a 0 Km/h para um trem que percorrer uma rota já alinhada em condições normais.

RGS17: Quando o sistema receber um comando de restrição de velocidade, ele deverá impor um limite máximo de velocidade em todos os circuitos de via envolvidos que estejam com um valor de código de velocidade acima desse limite.

RGS18: Só poderá haver abertura de um bloqueio como entrada de uma rota de chamada após um intervalo de tempo de 60 segundos, contado a partir da imposição de código de velocidade de 0 Km/h aos circuitos de via pertencentes à rota. A finalidade desse tempo é permitir ao trem, que eventualmente esteja trafegando em sentido oposto, parar completamente antes da abertura do bloqueio para a rota de chamada.

RGS19: Na ocorrência de violação de bloqueio, deverá ser imposto código de velocidade de 0 Km/h nos circuitos de via pertencentes à região invadida, com o fechamento imediato de todos os bloqueios abertos dessa região. A violação de bloqueio ocorre quando o trem não respeita o sinaleiro vermelho, ou o bloqueio fechado.

RGS20: Só poderá haver efetivação da inversão de sentido de tráfego se o bloqueio para o qual o(s) trem(s) se dirigirá(ão) não estiver sendo utilizado como saída de uma outra rota.

RGS21: A efetivação do modo de manutenção em uma região da via só pode ocorrer se não houver nenhuma rota alinhada na região. Rotas que estejam alinhadas devem entrar em processo de cancelamento. Os códigos de velocidade nos trechos da região em manutenção devem ser 0 Km/h e devem ser gerados perfis de frenagem nos trechos que antecedem os bloqueios de acesso à região.

RGS22: Uma vez efetivado o modo manutenção, nenhuma rota pode ser alinhada na respectiva região da via.

RGS23: A permanência de pessoas e a movimentação de veículos dentro de uma região em manutenção, devem estar regulamentados por procedimentos operacionais.

RGS24: Quando o trem estiver em operação manual, a distância segura entre dois trens deve ser garantida por procedimentos operacionais.

RGS25: Só poderá haver alinhamento de rota de chamada se não existir qualquer outra rota de chamada em sentido oposto ou proibição de bloqueio de saída.

RGS26: A simulação de ocupações na via não pode de forma alguma gerar condições que levem a situações inseguras.

RGS27: Um bloqueio deve permanecer fechado enquanto houver pelo menos uma requisição de proibição de abertura desse bloqueio.

Em função da arquitetura adotada pelo sistema de controle metro-ferroviário, novos requisitos deverão ser gerados visando cobrir aspectos de segurança decorrentes da arquitetura adotada.

REFERÊNCIAS BIBLIOGRÁFICAS

ARTIGOS

ATLEE, J.M.; GANNON,J. Stated-based model checking of event-driven systems requirements. **IEEE Transactions on Software Engineering**, v.19, n.1, p.24-40, Jan. 1993.

AVIZIENIS, A. Fault tolerance: The survival attribute of digital systems. **Procedures of the IEEE**, v.66, n.10, p.1109-25, Oct. 1978.

BUTLER, R. W.;FINELLI, G. B. The infeasibility of quantifying the reliability of life-critical real-time software. **IEEE Transactions on Software Engineering**, v.19, n.1, p.3-12, Jan. 1993.

COOLAHAN,J. E. JR.;ROUSSOPOULOS, N. Timing requirements for time-driven systems using augmented Petri nets. **IEEE Transactions on Software Engineering**, v.9, n.5, p.603-16, Sept. 1983.

COLL,D. C.;SHEIKH,A. U. H.;AYERS,R. G.;BAILEY,J. H. The communication system architecture of the North American Advanced Train Control System. **IEEE Transactions on Vehicular Technology**, v.39, n.3, p.244-55, Aug. 1990.

CRAIGEN,D.;GERHART,S.;RALSTON,T. Case study: Darlington nuclear generating station. **IEEE Software**, v.11, n.1, p.29-32, Jan. 1994.

CRAIGEN,D.;GERHART,S.;RALSTON,T. Case study: traffic alert and collision-avoidence system. **IEEE Software**, v.11, n.1, p.35-37, Jan. 1994.

EGNOT,J. M.;BBABICZ,G. M. Green line ATO builds on tried technology. **Railway Gazette International**, v.149, n.3, p.177-78, Mar. 1993.

FAULK,S.;BRACKETT,J.;WARD,P.;KIRBY,J. JR. The core method for real time requirements. **IEEE Software**, v. , n. , p.22-33, Sept. 1992.

FRASER,M. D.;KUMAR,K.;VAISHNAVI,V. K. Informal and formal requirements specification languages: bridging the gap. **IEEE Transactions on Software Engineering**, v.17, n.5, p. 454-66, May 1991.

GERHART,S.;CRAIGEN,D.;RALSTON,T. Experience with formal methods in critical systems. **IEEE Software**, v.11, n.1, p.21-8, Jan. 1994.

GERHART,S.;CRAIGEN,D.;RALSTON,T. Case study: Paris metro signaling system. **IEEE Software**, v.11, n.1, p.32-5, Jan. 1994.

GLORIA,A.;IMPALLOMENI,E. Software reliability and safety within the framework of European standardization for railway and subway applications. In: WORLD CONGRESS ON RAILWAY RESEARCH, Paris, 1994. **WCRR'94:** proceedings. Paris, SNCF, 1994. p.727-40.

GOPINATH,P.;BIHARI,T.;GUPTA,R. Compiler support for object-oriented real time software. **IEEE Software**, v.9, n.5, p.45-50, Sept. 1992.

HALANG,W. A.;KRAMER,B. J. Safety assurance in process control. **IEEE Software**, v.11, n.1, p.61-7, Jan. 1994.

HAMLET,D. Are we testing for true reliability?. **IEEE Software**, v.9, n.4, p.21-7, July 1992.

HECHT,H.;DUSSAULT,H. Correlated failures in fault tolerant computers. **IEEE Transactions on Reliability**, v.36, n.2, p.529-40, June 1987.

HOPE,R. Satellites track the Trains. **Railway Gazette International**, v.149, n.3, p.171-72, Mar. 1993.

HUDAK,J.;SUH,B.;H.;SIEWIOREK,D.;SEGALL,Z. Evaluation & comparison of fault-tolerant software techniques. **IEEE Transactions on Reliability**, v.42, n.2, p.190-204, June 1993.

HUSSEINY,A. A.;SABRI,Z. A.;ADAMS,S. K.;RODRIGUEZ,R. J. Automation of nuclear power plants. **Nuclear Technology**, v.90, p.34-48, Apr. 1990.

JAEGER P. K. Radio links in the channel tunnel. **Railway Gazette International**, v.149, n.4, p.255-56, Apr. 1993.

JAFFE,M. S.;LEVESON,N. G.;HEIMDAHL,M. P. E. M.;BONNIE E. Software requirements analysis for real time process control systems. **IEEE Transactions on Software Engineering**, v.17, n.3, p.241-58, Mar. 1991.

KADONO,T.;SATO,T.;IMAI,T. Recent train operation control. **Hitachi Review**, v.40, n.4, p.309-14, aug. 1991.

KANOUN,K.;KAÂNICHE,M.;BEUNES,C.;LAPRIE,J.C.;ARLAT,J. Reliability growth of fault-tolerant software. **IEEE Transactions on Reliability**, v.42, n.2, p.205-19, June 1993.

KECECIOGLU,D. Reliability education: a historical prespective. **IEEE Transactions on Reliability**, v.33, n.1, p.21-8, Apr. 1984.

KELLY,J. P. J.;MCVITTIE,T. I.;YAMAMOTO,W. I. Implementing design diversity to achieve fault tolerance. **IEEE Software**, v.8, n.4, p.61-71, July 1991.

KNIGHT,J. C.;LEVESON,N. G. An experimental evaluation of the assumption of independence in multiversion programming. **IEEE Transactions on Software Engineering**, v.12, n.1, p.96-109, Jan. 1986.

KNIGHT,J.;LITTLEWOOD,B. Critical task of writing dependable software. **IEEE Software**, v.11, n.1, p.16-20, Jan. 1994.

KUSAKARI,M;SAKUMA,M.;ISOBE,E.;TAKAOKA,T. On-Board train information control network systems. **Hitachi Review**, v.40, n.4, p.303-8, aug. 1991.

LALA,J. H.;HARPER,R. E. Architectural principles for safety critical real time applications. **Proceedings of the IEEE**, v.82, n.1, p.25-40, Jan. 1994.

LAMPORT,L.;SHOSTAK,R.;PEASE,M. The byzantine generals problem. **ACM Transactions on Programming Languages and Systems**, v.4, n.3, p.382-401, July 1982.

LAPRIE,J. C.;ARLAT,J.;BEOUNES,C.;KANOUN,K. Definition and analysis of hardware and software fault tolerant architectures. **Computer**, v.23, n.7, p.39-51, July 1990.

LAPRIE,J. C.;KNOUN,K.;BEOUNES,C.;KAÂNICHE,M. The KAT (knowledge action transformation) approach to the modeling and evaluation of reliability and availability growth. **IEEE Transactions on Software Engineering**, v.17, n.4, p.370-82, Apr. 1991.

LAPRIE,J.C.;KANOUN,K. X-Ware reliability and availability modeling. **IEEE Transactions on Software Engineering**, v.18, n.2, Feb. 1992.

LEE,I.;TANG,D.;IYER,R. K.;HSUEH,M. Measurement-Based evaluation of operating systems fault tolerance. **IEEE Transactions on Reliability**, v.42, n.2, p.238-49, June 1993.

LEVESON,N. G.;HARVEY,P. Analyzing software safety. **IEEE Transactions on Software Engineering**, v.9, n.5, p.569-79, Sept. 1983.

LEVESON,N. G. Software safety:why,what, and how. **Computing Surveys**, v.18, n.2, p.25-163, June 1986.

LEVESON,N. G.;CHA,S. S.;SHIMEALL,T. J. Safety verification of ADA programs using software fault trees. **IEEE Software**, v.8, n.4, p.48-59, July 1991.

LIN,K.J.;BURKE,E. J. Real time realities. **IEEE Software**, v.9, n.5, p.13-15, Sept. 1992.

LUNDGREN,J. R. Cooperation in railway research: the association of american railroads approach and experience. In: WORLD CONGRESS ON RAILWAY RESEARCH, Paris, 1994. **WCRR'94:** proceedings. Paris, SNCF, 1994. p.553-8.

LYU,M. R.;HE,Y. T. Improving the N-versions programming process through the evolution of a design paradigm. **IEEE Transactions on Reliability**, v.42, n.2, p.179-89, June 1993.

MASSUR,A. Studies on some alternative arquitectures for fault-tolerant automotive multiplexing network systems. **IEEE Transactions on Vehicular Tchnology**, v.40, n.2, p.501-10, May 1991.

MEYER,J. F.;PHAM,H. Fault-tolerant software- guest editors'prolog. **IEEE Transactions on Reliability**, v.42, n.2, p.177-8, June 1993.

MOJDEBAKHSH,R.;TSAI,W.;SEKHAR,K.;ELLIOTT,L. Retrofitting software safety in an implantable medical device. **IEEE Software**, v.11, n.1, p.41-50, Jan. 1994.

MÜNCH,K. H. The promise of computer integrated railroading. **Siemens Review, R&D Special**, p.11-15, Spring 1993.

NATARAJAN,S.;ZHAO,W. Issues in building dynamic real time systems. **IEEE Software**, v.9, n.5, p.16-21, Sept. 1992.

ORTH,C. L. Managing R&D in the United States: federal railroad administration prespective, In: WORLD CONGRESS ON RAILWAY RESEARCH, Paris, 1994. **WCRR'94:** proceedings. Paris, SNCF, 1994. p.451-5.

PERRY,T. S. Improving the world's largest, most advanced system. **IEEE Spectrum**, v.28, n.2, p.22-36, Feb. 1991.

PFLEEGER,S. L. Measuring software reliability. **IEEE Spectrum**, v.29, n.8, p.56-60, Aug. 1992.

POSPISCHIL,G.;PUSCHNER,P.;VRCHOTICKY,A. Developing real time tasks with predictable timing. **IEEE Software**, v.9, n.5, p.35-44, Sept. 1992.

PUCCI,G. A New approach to the modeling of recovery block structures. **IEEE Transactions on Software Engineering**, v.18, n.2, p.159-67, Feb. 1992.

RAILWAYS European train control system. **Modern Railways**, p.546-8, Sept. 1994.

RANGANATHAN,A.;UPADHYAYA,S. J. Performance evaluation of rollback-recovery techniques in computer programs. **IEEE Transactions on Reliability**, v.42, n.2, p.220-6, June 1993.

RAO,V. P.;VENKATACHALAM,P.A. Microprocessor based railway interlocking control with low accident probability. **IEEE Transactions on Vehicular Technology**, v.35, n.3, p.141-7, Aug. 1987.

RAVN,A. P.;RISCHEL,H;HANSEN,K. M. Specifying and verifying requirements of real-time systems. **IEEE Transactions on Software Engineering**, v.19, n.1, p.41-55, Jan. 1993.

RUSHBY,J. M.;HENKE,F. Formal verification of algorithms for critical systems. **IEEE Transactions on Software Engineering**, v.19, n.1, p.13-23, Jan. 1993.

RUSHBY,J. Critical system properties: survey and taxonomy. **Reliability Engineering & System Safety**, v.43, n.2, p.189-219, aug. 1994.

SAEED, A.;LEMOS,R.;ANDERSON,T. The role of formal methods in the requirements analysis of safety-critical systems: a train set example. In: 21ST IEEE INTERNATIONAL SYMPOSIUM ON FAULT-TOLERANT COMPUTING, Montreal, 1991. **FTCS-21**: proceedings. Montreal, IEEE, 1991. p.478-85.

SCHNEIDEWIND,N. F.;KELLER,T. W. Aplying reliability models to the sace Shuttle. **IEEE Software**, v.9, n.4, p.28-33, July 1992.

SCOTT,R. K.;GAULT,J. W.;MCALLISTER,D. F. Fault tolerant software reliability modeling. **IEEE Transactions on Software Engineering**, v.13, n.5, p.582-92, May 1987.

SHAW,A. C. Communicating real time state machines. **IEEE Transactions on Software Engineering**, v.18, n.9, p.805-16, Sept. 1992.

SHEIKH,A. U. H.;COLL,D. C.;AYERS,R. G.;BAILEY,J. H. ATCS: Advanced train control system radio data link design consideration. **IEEE Transactions on Vehicular Technology**, v.39, n.3, p.256-62, Aug. 1990.

SHELDON,F. T.;KAVI,K. M.;TAUSWORTHE,R.C.;YU,J.;BRETTSCHNEIDER,R.;EVERETT,W. W. Reliability measurement: from theory to practice. **IEEE Software**, v.9, n.4, p.13-20, July 1992.

SHIN,K. G.;LEE,Y. H. Error detection process: model, design and its impact on computer performance. **IEEE Transactions on Computer**, v.33, n.6, p.130-47, June 1984.

SHIN,K. G.;DOLTER,J. W. Alternative majority voting methods for real time computing systems. **IEEE Transactions on Reliability**, v.38, n.1, p.58-64, Apr. 1989.

SHIN,K. G.;KANDLUR,D. D.;KISKIS,D. L.;INDIRESAN,A. A distributed real time operating system. **IEEE Software**, v.9, n.5, p.58-68, Sept. 1992.

SINGH,A. D. Fault tolerant systems. **Computer**, v.23, n.7, p.16-17, July 1990.

SOSNOWSKI,J. Transient fault tolerance in digital systems. **IEEE Micro**, v.14, n.1, p.24-35, Feb. 1994.

STANKOVIC,J. A. Misconceptions about real time computing. **Computer**, v.21, n.10, p.10-19, Oct. 1988.

STANKOVIC,J. A.;RAMAMSITHAM,K. The spring kernel: a new paradigm for real time systems. **IEEE Software**, v.8, n.3, p.62-72, May 1991.

STRIGINI,L. Considerations on current research issues in software safety. **Reliability Engineering and Systems Safety**, v.43, n.2, p.177-88, feb. 1994.

TAI,A. T,;MEYER,J. F.;AVIZIENIS,A. Performability enhancement of fault-tolerant software. **IEEE Transactions on Reliability**, v.42, n.2, p.227-37, June 1993.

TAKAOKA,T.;KAWASHIMA,H.;TASHISO,K. High speed and high congested train operation in railway systems. **Hitachi Review**, v.40, n.4, p.297-302, aug. 1991.

VERNAZZA,G;ZUNINO,R. A distributed intelligence methodology for railway traffic control. **IEEE Transactions on Vehicular Technology**, v.39, n.3, p.263-70, Aug. 1990.

ZAHEDI,F.;ASHRAFI,N. Software reliability allocation based on structure, utility, price and cost. **IEEE Transactions on Software Engineering**, v.17, n.4, p.345-56, Apr. 1991.

ZHAO,W.;RAMAMRITHAM,K.;STANKOVIC,J. A. Scheduling tasks with resource requirements in hard real time systems. **IEEE Transactions on Software Enginerring**, v.13, n.5, p.564-77, May 1987.

WARD,P. T. The transformation schema: an extension of the data flow diagram to represent control and timing. **IEEE Transactions on Software Engineering**, v.12, n.2, p.198-210, Feb. 1986.

WEIS,R. S. Moving block signalling offers cost savings. **Developing Metros**, p.19-20, mes. 1992.

WENSLEY,J. H.;LAMPORT,L.;GOLDBERG,J.;GREEN,M. W.;LEVITT,K. N.;MELL IAR,P.;SHOSTAK,R. E.;WEINSTOCK,C. B. SIFT: Design and analysis of a fault tolerant computer for aircraft control. **Proceedings of the IEEE**, v.66, n.10, p.1240-55, Oct. 1978.

WILLIAMS,T. Real time operating systems struggle with multiple tasks. **Computer Design**, v.29, n.19, p.92-108, Oct. 1990.

WILLIAMS,L. G. Assessment of safety critical specifications. **IEEE Software**, v.11, n.1, p.51-60, Jan. 1994.

LIVROS

JOHNSON,B. W. **Design and fault-tolerant digital systems**. Charlottesville Addison-Wesley, 1989.

KAPUR,K.C.;LAMBERSON,L.R. **Reliability in engineering design**. Detroit John Wiley & Sons, 1977.

KECEIOGLU,D. **Reliability engineering handbook**. New Jersey Prentice Hall, 1991.

LEVENSON,N. G. **SAFEWARE: system safety and computers**. Washington Addison-Wesley, 1995.

RICH,E. **Artificial intelligence**. New York MCGraw-Hill Book Company, 1983.

ROOK,P. **Software reliability handbook**. London Elsevier Applied Science, 1990.

SIEWIOEREK,D. P.;SWARZ,R. S. **The theory and practice of reliable system design**. Bedford Digital Press, 1974.

NORMAS E GUIAS

[THE CHANNEL TUNEL-94] A Safety Case, Eurotunnel Exibition Center, 1994.

[NASA-STD-2202-93]Software Formal Inspection Standard.

[NASA-GB-A302-93]Software Formal Inspection Guidebook.

[NASA-STD-2201-93]Software Assurance Standard.

[ANSI/IEEE 830-84]Guide to Software Requirements Specifications.

[FDTE 542.401.AS.001] Determinação, Análise e Classificação das Funções de Segurança do Sistema Paraíso-Clínicas, março 1991.