



Validação de Sistemas Críticos

MC646 - R. Anido



Perspectivas da Validação

- Validação da Confiabilidade
 - **A confiabilidade medida do sistema está de acordo com a especificação?**
 - **A confiabilidade do sistema é suficiente para os usuários?**
- Validação de Segurança
 - **O sistema sempre opera de maneira que acidentes não ocorrem (ou suas consequências são minimizadas)?**
- Validação de Proteção
 - **O sistema e seus dados estão seguros contra ataques externos?**



Técnicas de Validação

- Técnicas Estáticas

- **Revisão de projeto, inspeção de código**
- **Provas e argumentos matemáticos**

- Técnicas

- **Teste estatístico**
- **Teste baseado em cenário**
- **Verificação (checking) de run-time**

- Validação de Processo

- **Processos de desenvolvimento que minimizem a probabilidade de erros que possam comprometer a confiabilidade do sistema.**



Validação Estática para Segurança

- Demonstração da segurança através de testes é difícil (deve mostrar o que o sistema faz em todas as situações)
- Revisões/inspeções devem ser suplementadas com foco em verificar que situações inseguras não acontecem



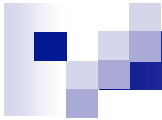
Revisões de segurança

- Funcionalidade correta sendo implementada
- Estrutura compreensível, importante para manutenção
- Algoritmos e estruturas de dados compatíveis com especificação
- Adequação para testabilidade



Exemplos de boas práticas

- Software tão simples quanto possível
- Uso de “information hiding” sempre que possível
- Uso de técnicas de tolerância a falhas (mas não são infalíveis!)



Análise baseada em riscos

- Garantia de segurança efetiva depende de identificação de riscos
- Segurança pode ser conseguida por
 - **Evitar riscos**
 - **Evitar acidentes**
 - **Proteção de sistemas**
- Revisões de segurança devem demonstrar que uma ou mais destas técnicas foram aplicadas para todos os riscos identificados



Segurança de sistemas

- É prática normal que sistemas críticos tenham um “formal safety case”
- “Safety Case” apresenta uma lista de argumentos, baseado em riscos detectados, do por que esses riscos não causarão um acidente
- Argumentos: prova formal, filosofia do projeto, provas de segurança, fatores do processo de desenvolvimento, etc.



Métodos Formais e sistemas críticos

- Em alguns países (Inglaterra), é mandatório para o desenvolvimento de alguns tipos de sistemas críticos
- A efetividade não é consenso
- Ainda muitos problemas



Provas de segurança

- Objetivo é mostrar que sistema não pode atingir estado inseguro
- Mais fracas do que prova de correção (conformidade com especificação)
- Geralmente baseadas em provas por contradição
 - **Assumir que um estado inseguro tenha sido alcançado**
 - **Mostrar que isto gera uma contradição com o código do programa**



Construção de uma prova de segurança

- Estabeleça as condições seguras para término do componente/programa
- Começando pelo final, ande de trás para frente até identificar todos os caminhos que levam ao término do programa
- Assuma que a condição de saída é falsa
- Mostre que, para cada caminho que leva à saída, as atribuições feitas contradizem a suposição de um término inseguro



Sistema de Detecção de Gás

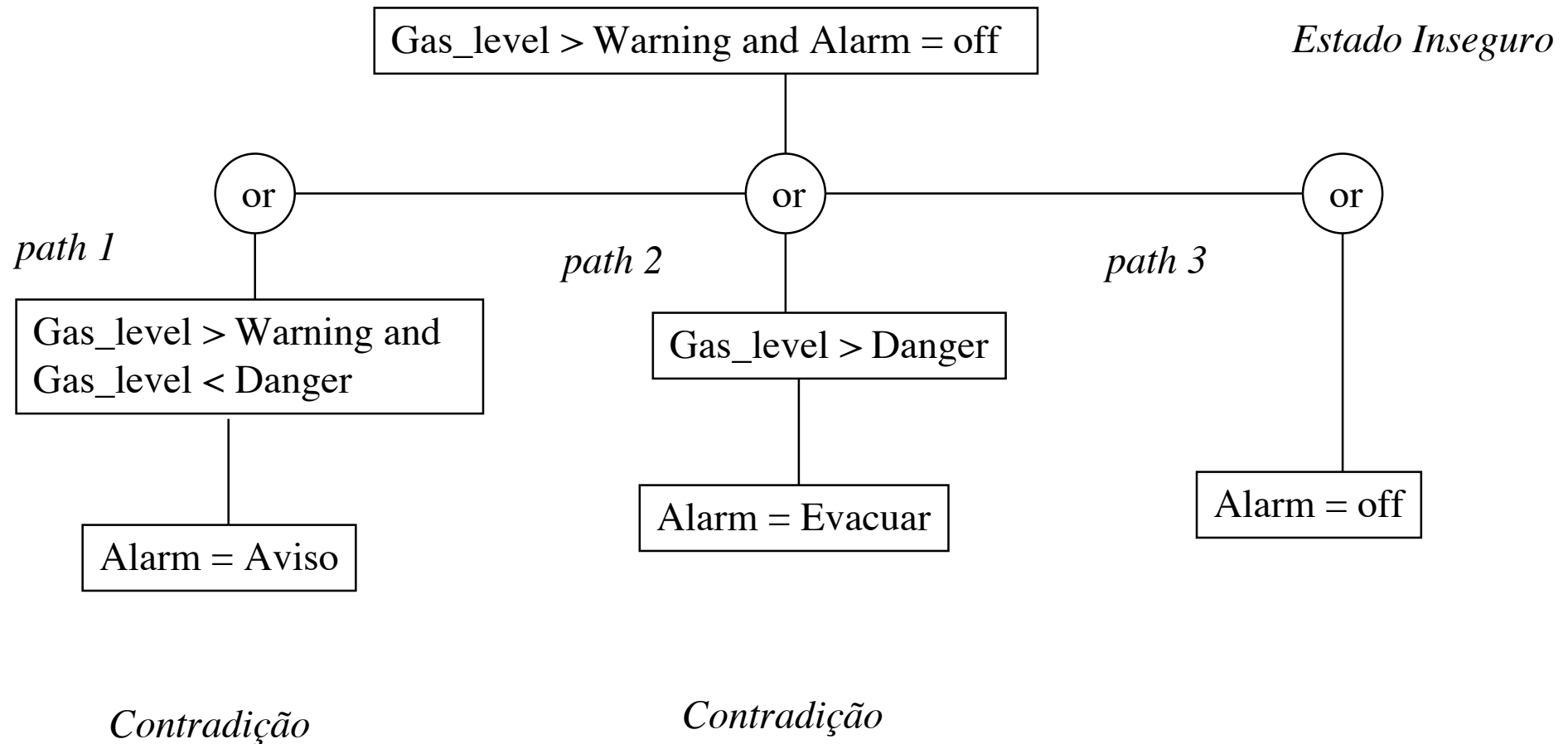
- Sistema para avisar sobre presença de gás tóxico.
Consiste de um sensor, um controlador e um alarm
- Dois níveis de gás são perigosos
 - **Nível de atenção - sem perigo imediato, mas tome medidas para corrigir**
 - **Nível de evacuação - perigo imediato, evacuar a área**
- O controlador captura amostras de ar, computa o nível de gás e decide se alarme deve ser ativado



Controlador do sistema de gás

```
Gas_level: GL_TYPE ;  
loop  
    -- Toma 100 amostras de ar  
    Gas_level := 0.000 ;  
    for i in 1..100 loop  
        Gas_level := Gas_level + Gas_sensor.Read ;  
    end loop ;  
    Gas_level := Gas_level / 100 ;  
    if Gas_level > Warning and Gas_level < Danger then  
        Alarm := Warning ; Wait_for_reset ;  
    elsif Gas_level > Danger then  
        Alarm := Evacuate ; Wait_for_reset ;  
    else  
        Alarm := off ;  
    end if ;  
end loop ;
```

Argumento Gráfico





Verificação de condições

Gas_level < Warning	Path 3	Alarm = off (Contradiction)
Gas_level = Warning	Path 3	Alarm = off (Contradiction)
Gas_level > Warning and Gas_level < Danger	Path 1	Alarm = Warning (Contradiction)
Gas_level = Danger	Path 3	Alarm = off
Gas_level > Danger	Path 2	Alarm = Evacuate (Contradiction)

Código é incorreto.

Gas_level = Danger não causa Alarme ser ligado



Pontos importantes

- Sistemas com aspectos de segurança devem ser desenvolvidos de maneira mais simples possível, utilizando técnicas ‘seguras’ de desenvolvimento.
- Provas de segurança são mais simples do que provas de consistência ou correção.



Técnicas de Validação Dinâmicas

- Testes - analisar o sistema fora de seu ambiente operacional
- Verificação de run-time - analisar se o sistema está operando dentro de um “envelope” de confiabilidade



Validação de Confiabilidade

- Exercitar o programa para verificar se atingiu o nível requerido de confiabilidade
- Não é o mesmo que processo de teste para defeitos
- Testes estatísticos devem ser usados, baseados em amostra real/simulada de uso



Processo de Validação

- Estabelecer um ‘perfil operacional’ para o sistema
- Construir dados de teste que reflitam o perfil operacional
- Testar o sistema e observar o número de falhas e os tempos de ocorrência
- Computar a confiabilidade após número significativo de falhas terem sido observadas



Perfil operacional

- Conjunto de dados de teste com frequência igual à frequência de entradas em uma operação ‘normal’ do sistema
- Pode ser gerado automaticamente ou (mais frequentemente) depende de suposições sobre o padrão de uso do sistema
- Difícil de prever entradas “improváveis”...
- Perfil pode se modificar com o tempo...