Redundância de SW + (Precondições, Pós condições + Assertions + Manipulação de excessão +

Há duas principais abordagem para detectar erros causados por erros no projeto.

(1) Teste de Aceitação: execução de asserções formais sobre o processo sendo executado

(2) Diversidade de Projeto: ~~Diversidade~~ Variantes de software podem ser comparadas. (produzidas a partir de uma mesma especificação)

Métodos para Tolerância de Erros no Software (Variantes de SW).

Os métodos mais conhecidos são:

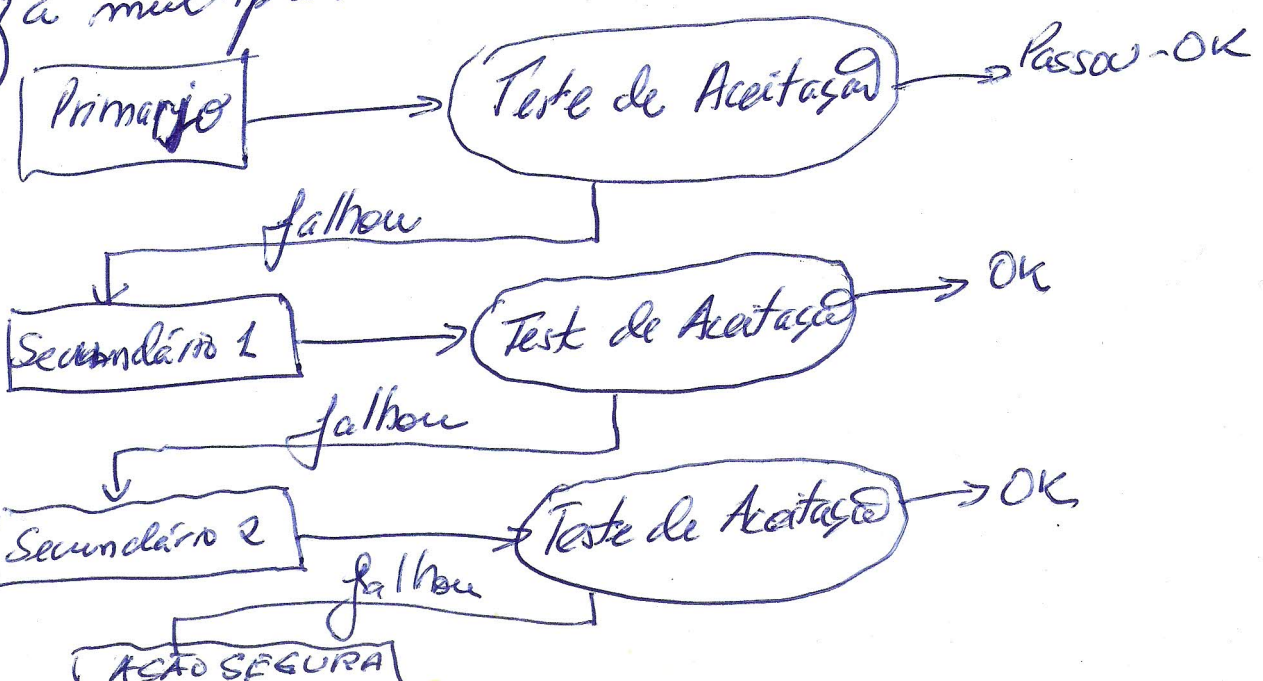
a) Recovery Block (RB)

b) N-Version Programming (NVP)

c) N Self checking Programming (NSCP)

Recovery Block (RB)

Utiliza multiplas variantes de SW.



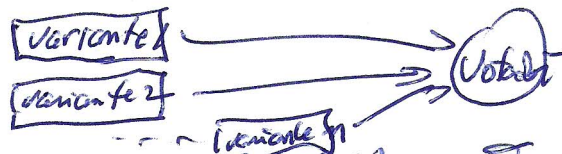
Quando o teste de aceitação falha o sistema deve retornar ao estado imediatamente anterior à execução da variante de software em questão. (rolled back)

O sucesso do método Recovery Block (RB) depende basicamente de:

a) Não haver bugs correlatos entre as diversas variantes de SW.

b) Qualidade do Teste de Aceitação.

N- Version Programming



Neste método de tolerância a erros de sw são usadas N equipes independentes de programadores que desenvolvem variantes de sw a partir de uma mesma especificação.

As N variantes rodam em paralelo e suas saídas são votadas. A esperança é que os programadores terão uma probabilidade baixa de falharem para as mesmas entradas.

Dificuldades neste tipo de método:

1) Problema de comparação consistente

Inconsistências podem ocorrer na comparação se a precisão exigida é muito alta e as variantes de sw diferem em pouco da outra.

Este problema pode ocorrer, por exemplo, quando as Variantes devem comparar com um determinado threshold.

A frequência e o tempo de duração que essas inconsistências podem ocorrer dependem da natureza da aplicação.

2) Independência das Variantes

Essa independência pode ser comprometida devido a diversos fatores:

a) Especificação comum.
[Uma redução é usar diversidade de especificações,] depois
[expressas em diferentes formalismos.

b) Devido a dificuldades intrínsecas do domínio de aplicação, pode haver uma maior dificuldade de implementação para certo conjunto de dados de entrada, aumentando a correlação entre as variantes de SW.

c) Utilização de algoritmos comuns para determinadas regiões do espaço de entrada.

d) Fatores culturais: programadores que são treinados para pensarem de maneira similar e cometerem erros similares.

e) Plataformas comuns de SW e HW.

Algumas formas de contornar essas dificuldades:

a) ⊛

b) Usar linguagens de programação diversas em cada variante

c) Usar ferramentas de desenvolvimento e compiladores diferentes em cada variante

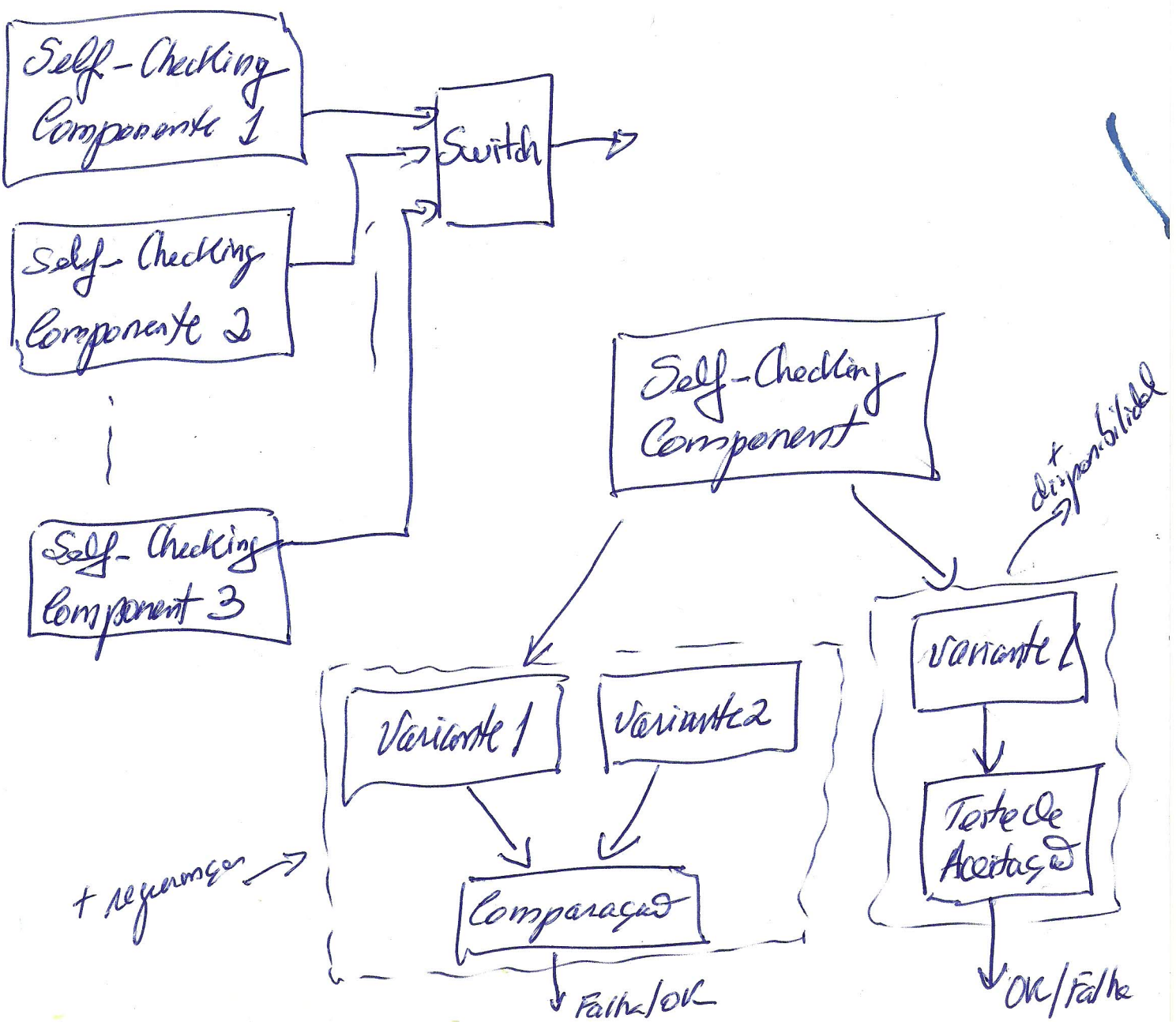
d) Usar equipes com diversidade cognitiva (pensam de forma diferente)

N Self Checking Programming (NSCP)

A tolerancia a erro de sw é alcançada através da execução paralela de pelo menos 2 componentes com auto-teste (self-checking).

Em cada execução apenas um componente age, enquanto os demais permanecem em "hot standby".

O processamento do erro é feito através da detecção dos erros e chaveamento dos componentes.



Arquiteturas de HW/SW Tolerante a Erros (Fault-Errors)

Estas arquiteturas toleram erros de hw e sw enfatizando a dependência entre eles. Serão apresentadas duas categorias de arquitetura: tolerante a "single fault" e a "two consecutive faults"

Há dois aspectos importantes:
- nº de variantes
- nível em que a tolerância ao erro é aplicado

O nº de variantes está diretamente relacionado ao número de erros a serem tolerados.

O nível de tolerância ao erro compreende:

1. O quanto o sistema deve ser decomposto em componentes para ser diversificado. (Granularidade).
   - componentes menores permitem um melhor controlo dos algoritmos de decisão
   - componentes maiores auxiliam na diversidade
2. Quais camadas devem ser diversificadas? (sw aplicativo, comunicação, básico, hw)

Princípios Estruturais para a Definição da Arquitetura

Considerando defeitos de HW e SW devemos distinguir áreas de confinamento desses defeitos, tanto para HW quanto para SW.

HECAs: Áreas de Confinamento de erro de HW

SECAs: Áreas de Confinamento de erro de SW

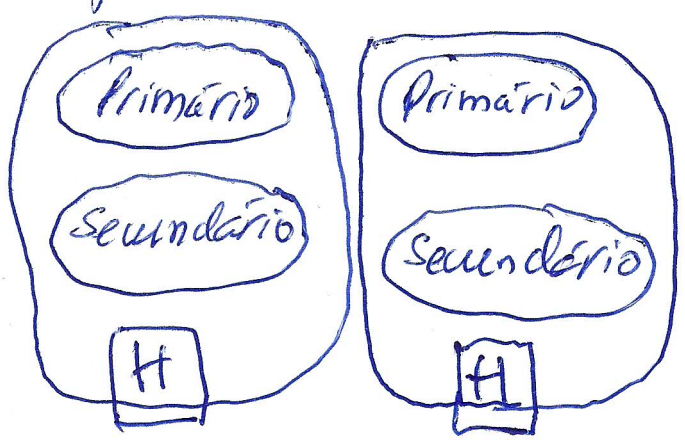
X/z/y/

→ Nº de erro de SW tolerado

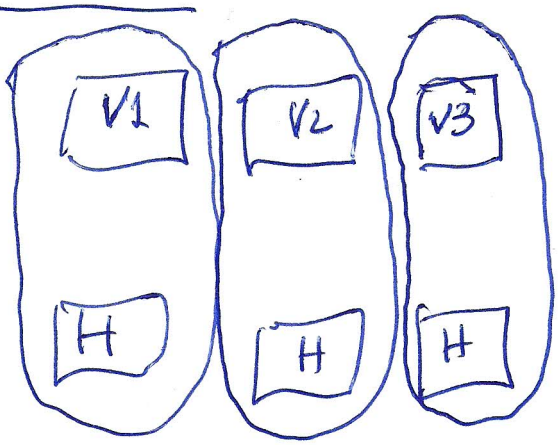
→ Nº de erro de HW tolerado

→ Modelo de tolerância a erro de SW: RB/NVP/NSCP

Arquiteturas Tolerantes a um Único Erro

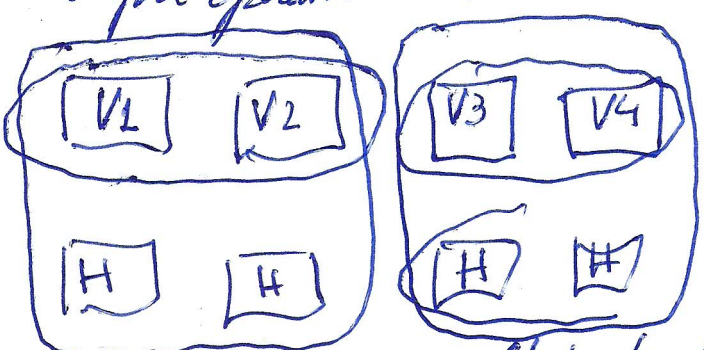


RB/1/1



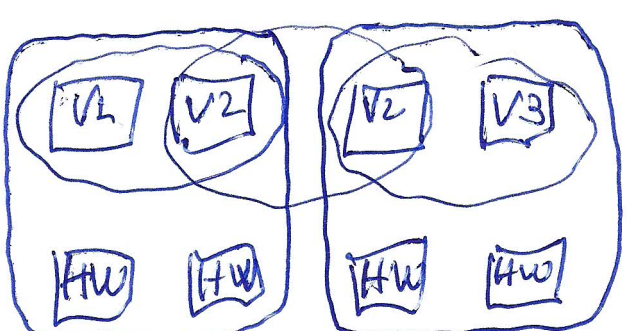
NVP/1/1

Obs: Os HWs operam em Hot Standby e sempre operam a mesma Variante



NSCP/1/1 (Hot-standby)

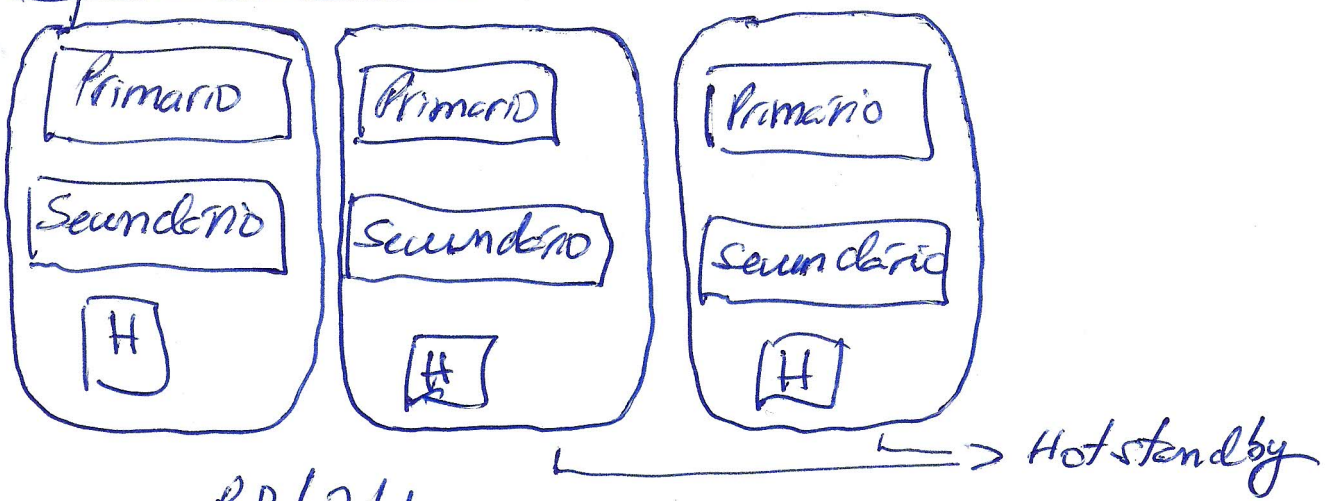
↳ Detecção por comparação (Airbus A-320)



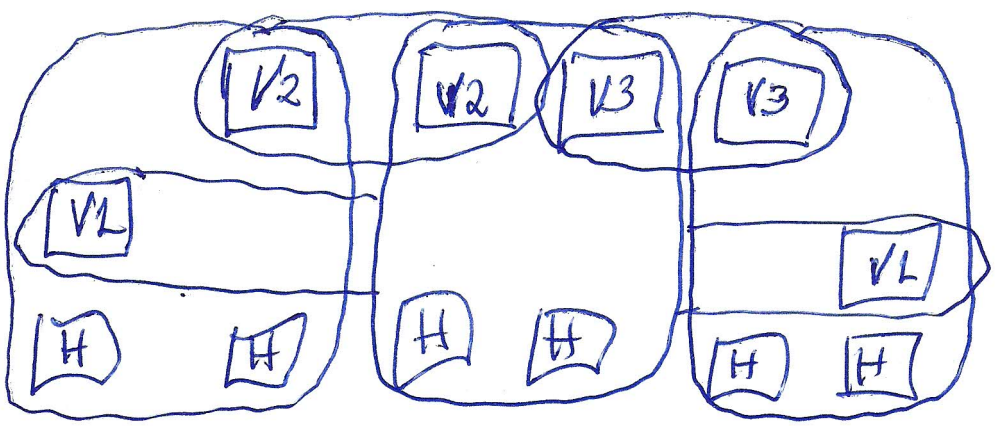
NSCP/1/1/m (modified)

Se o V2 era causa discrepância nos 2 blocos ⇒ Saída V1 x V3

Arquiteturas Tolerantes a Dois Erros Consecutivos

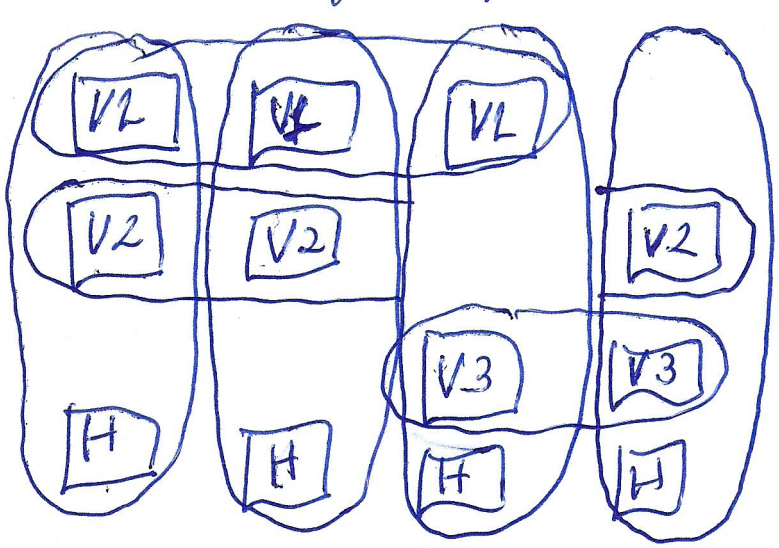


RB/2/1



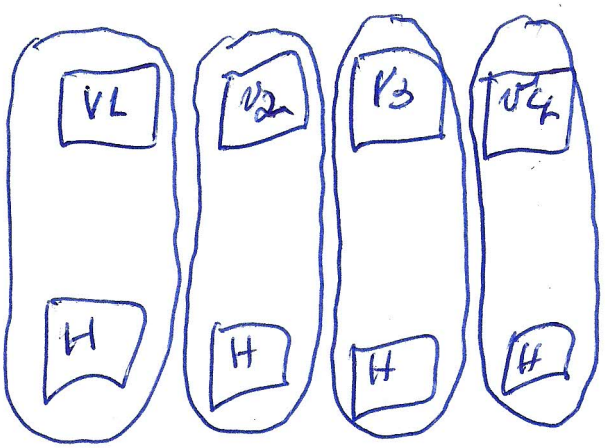
NSCP/2/1 (extensão do NSCP/1/1/m)

↳ deteção por comparação



NVP/2/1

Votação Majoritária 2 de 3



NVP/2/2

Votação Majoritária 2 de 3

Em cada canal uma variante é ativa e a outra hot-standby