Sumário

- Motivação.
- Fiabilidade, falha e faltas.
- Prevenção e tolerância.
- Programação N-versões.
- Redundância dinâmica por software.
- Blocos de recuperação.
- Excepções.
- Confiança no funcionamento.

Motivação para a tolerância a faltas

- Segurança de pessoas e bens:

  Aviónica.

  Controlo de central nuclear.

  Sistemas médicos.
- Garantir um serviço "seguro", mesmo na presença de faltas.

Causas das falhas

- Especificação inadequada:

  A especificação não reflecte os requisitos ou condições operacionais.
- Erros de concepção ou programação:
- Falhas de componente:

  Memória, bus, CPU, etc.
- Interferência no sistema de comunicação:

  Perda de mensagens, mensagens adulteradas, partições na comunicação.

Fiabilidade e falha

- Fiabilidade:

  Comportamento de acordo com a especificação.
- Falha:

  Desvio do comportamento especificado.

Falta, erro e falha.

- Falha: desvio do comportamento.

  O PC não arranca.

- Erro:

  O disco não responde.

- Falta:

  Cabo alimentação quebrado.

Tipos de faltas: comportamento no tempo

- Faltas temporárias.

  Falha que ocorre só durante um determinado período, geralmente devido a influência externa (ruído na comunicação).

- Faltas permanentes.

  Que permance até ser reparada (cabo solto).

- Faltas intermitentes.

  Que ocorre de tempos a tempos (por exemplo, sobreaquecimento com paragem).

Tipos de faltas: gravidade

- Benignas.

  Cujas conseqências são fácilmente controláveis

- Maliciosas.

  Geradas por um inimigo.

- Arbitrárias.

  Tudo pode acontecer.

Meios para obter confiança no funcionamente

- Supressão (“avoidance”) de faltas. *Ex. debug, simulação, validaçao*
- Previsão (“forecasting”) de faltas. *Ex. injecção.*
- Prevenção (“prevention”) de faltas. *Ex. self-check.*
- Tolerância (“tolerance”) a faltas. *Ex. votadores.*

Prevenção de falhas

- Uso de componentes fiáveis:

  Com maior intervalo entre falhas.

- Técnicas apropriadas de montagem:

  Reduzir vibrações, desgaste, etc.

- Protecção do hardware:

  Interferência electromagnética, fogo, choque, etc.

Prevenção de falhas (continuação)

- Especificação rigorosa.

- Uso de técnicas formais:

  De concepção e validação.

- Linguagens de programação adequadas:

  Forte verificação de tipos, modularidade.

- Ambientes de desenvolvimento adequados:

  Simuladores, programas de teste, etc.

Teste (remoção de faltas)

- Limitações:

  Permite descobrir faltas, mas não provar que elas não existem.

  Nunca cobre todos os casos.

  Por vezes é impossível testar em condições realistas.

Tolerância a faltas

- Completa:

  O sistema continua operacional apesar da ocorrência (de um número pré-determinado) de faltas.

- Degradação não abrupta:

  O sistema continua a fornecer "serviços mínimos".

- Paragem-segura:

  O sistema interrompe o seu funcionamento num estado seguro.

  Exemplo: parar os combóios.

Estratégia

- Optimista:

  O sofware é correcto.

  Os modos de falha dos componentes são conhecidos.

  As interacções entre o sistema e o ambiente são previsíveis.

- Péssimista:

  Falhas arbitrárias.

  Falhas de software.

- Custo/benefício.

Redundância: mascaramento

- Redundância tripla (*Triple Modular Redundancy (TMR)*):

  Três componentes idênticos e um circuito votador.

  Assume-se que os componentes não dão todos o mesmo erro.

- N-redundância:

  Semelhante com N componentes para tolerar mais faltas.

Redundância: detecção e recuperação

- Detectar o erro:

  Bastam duas.

- Corrigir o erro:

  Componente adicional.

Programação n-versões

- Várias versões do mesmo programa.
- Comparação de resultados.

  Votação: utilização de um programa comparador.

- Realimentação das réplicas com o resultado da comparação:

  Continuar, abortar uma ou mais réplicas, alterar os critérios de votação.

Programação n-versões: problemas

- Especificação não ambígua:

  Para poder comparar e votar os resultados.

- Independências do desenvolvimento:

  Para não terem todos o mesma falta.

- Financiamento:

  Para pagar a várias equipas de programadores.

Programação n-versões: votadores

- Nem sempre os resultados podem ser comparados bit a bit:

  Exemplo: duas leituras de um valor podem estar correctas embora ligeiramente diferentes, por serem feitas em pontos ligeiramente afastados ou em instantes diferentes.

- Votação inexacta:

  Ver se os valores estão dentro de um intervalo em relação à média dos valores.

Programação n-versões: comparação consistente

- Quando o programa testa se uma determinada leitura atingiu um valor limite e toma uma acção especial.
- Se o valor real está no limite.

  Cada réplica pode tomar uma decisão diferente e correcta.

Programação n-versões: princípios

- Especificação inicial correcta:

  Caso contrário, todas as versões estarão erradas.

- Concepção independente:

  As equipas não devem trocar ideias.

- Finaciamento suficiente.

Redundância passiva (dinâmica)

- Detecção do erro.
- Avaliação e limitação dos danos.
- Recuperação.
- Tratamento da falta e continuidade do serviço.

Detecção de erros (pelo ambiente)

- Instrução ilegal.
- Acesso fora do segmento.
- Ponteiro nulo utilizado.
- Divisão por zero.

Detecção de erros (pela aplicação)

- Replicação (realização da mesma operação múltiplas vezes).
- Testes de actividade.
- Realizar a operação inversa (prova-dos-nove).
- Codificação (códigos detectores de erros).
- Verificação da razoabilidade (ver se os valores estão dentro dos limites previsíveis).
- Verificação da estrutura (listas ligadas, etc).

Avaliação e limitação dos danos

- Firewall.
- Decomposição modular:

  Definição das interfaces entre os componentes limita o modo como os erros se propagam.

  Transações ou acções atómicas: conjuntos de acções que se realizam na totalidade ou nenhuma se realiza.
- Mecanismos de protecção:

  Espaços de endereçamento.

Recuperação

- Para a frente:

  Redundância no espaço: permite continuar sem as réplicas em falta.

- Para trás:

  Redundância no tempo: voltar a fazer de novo.

  A partir de que ponto?

Recuperação para trás

- Guardar periodicamente o estado do sistema.

  Salvaguarda (checkpointing).

- Voltar a executar tudo desde o último ponto de salvaguarda.
- É necessário voltar atrás também com os componentes que entretanto "viram" resultados que foram anulados.

  Apróximação "Esqueçam o que eu disse! Vamos voltar atrás".

- Efeito dominó:

  Salvaguardas coerentes.

Tratamento das faltas

- Reparação da maquinaria:
- Substituição do software.
- Como realizar estas operações "em funcionamento"?

Blocos de recuperação

- Conjunto de intruções limitados por:

  Um ponto de entrada designado por ponto de recuperação.

  E por um teste de aceitação.

- Testes de aceitação:

  Verifica se o resultado do bloco é aceitável.

- Caso falhe, especificam-se blocos alternativos:

  Que também devem passar no teste de aceitação.

Bloco de recuperação: exemplo

```
ensure <teste de aceitacao>
by
 lista de intrucoes primarias
else by
 primeira alternativa
else by
 segunda alterantiva
...
else error
```


N-versões e blocos de recuperação: comparação

- Estática (N-versões) e dinâmica (blocos de recuperação).
- Sobrecarga:

  N-versões: mais recursos.

  Blocos de recuperação: salvaguardas.
- Concepção: *Sujeitas ao mesmo tipo de erros.*
- Atomicidade: *Blocos de recuperação não cancelam acções com o exterior (estas podem ser mascaradas com N-versões).*

Excepções.

- Excepção:

  Acorrência de um erro.
- Levantar a excepção:

  Indicar a detecção do erro.
- Tratamento da excepção:

  Insruções (geralmente de cariz correctivo).

Excepções: internas e de interface

- Excepção interna:

  Devido a falha do componente.
- De interface:

  Devido à falha de outros componentes necessários para a execução do serviço.

Medição da fiabilidade.

- Função de fiabilidade:

  Propabilidade de estar a funcionar continuamente por um período de tempo t: $R(t) = e^{-\lambda t}$.

- Tempo entre falhas.

  Mean Time Between Failures (MTBF): $1/\lambda$.

- Disponibilidade:

  Probabilidade de estar acessível num determinado instante.

Segurança (safety).

- Distinção entre “safety” e “security”.
- Evitar as falhas catastróficas.

  Independentemente de continuar ou não a fornecer o serviço.

Confiança no funcionamento.

- Fiabilidade (reliability).
- Segurança (safety).
- Protecção contra intrusos (secure).

Confiança no funcionamento (continuação)

- Medidas de:

  Disponibilidade, fiabilidade.

- Limitações:

  Faltas, erros e falhas.

- Meios:

  Validação (previsão e remoção).

  Gestão (prevenção e tolerância).

Resumo

- Prevenção e tolerância.
- Técnicas de tolerância a faltas.
- Confiança no funcionamento.