Test and dependability 1: Problem handling during operation

Prof. G. Kemnitz

November 2, 2026

Contents

1 Dependability **6**
1.1 Service-Modell 6
1.2 Availability 8
1.3 Reliability 13
1.4 Safety (security) 17

2 Problems under Use **24**
2.1 Monitoring 24
2.2 Format Checks 26
2.3 Value checks, diversity 28
2.4 Time Monitoring 33
2.5 Correction 34
2.6 Automated/Manual 38

1.2 Foreword

The volume of specialist text grows exponentially over the years, whilst the time available for teaching remains constant. A major challenge for education.

»Test and dependability« summarises the fundamentals of diagnosis, test, maintenance, ... from this point of view in a new way.

Fields of knowledge are structured hierarchically. Knowledge modules can be thought of as stepping stones in an unfamiliar wilderness. Each stone mastered opens up new perspectives for the learner, paving the way to the next one.

Testing and dependability are central to computer science. IT systems are only useful if they can be trusted to

- deliver results on demand (availability),
- ensure that the results are correct (reliability) and
- prevent the system from causing catastrophic damage (security).

By choosing the right stepping stones (definitions, models, etc.), drastically simplifies the presentation of important concepts and, as a result, reduces the time needed to convey that knowledge:

> Reliability tends, if all detectable faults are eliminated, to be proportional to the number of tests and inversely proportional to the number of undetected faults.
>
> Where hazards can be ruled out for identified problems, safety is proportional to reliability and can be enhanced either through more testing, fault prevention, improved runtime checks, additional safety equipment or longer maturity periods.

In this conceptual framework, design and manufacturing are processes in which faults arise. Tests act as filters that reveal problems. Based on this, models are developed for qualitative and quantitative assessments.

The course covers topics ranging from self-testing and hardware fault tolerance to a vision for solving current reliability and security issues in software.

1.3 Organisation

Lecture web page: http://techwww.in.tu-clausthal.de/TestVerl

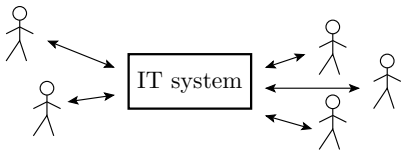
- Slide sets, handouts, homework, video recordings
- Submission of homework by e-mail to ha-tv@in.tu-clausthal.de as PDF. Deadlines see web page.
- Homework will be graded and returned. Additional publication of the number of points on the website.
- Exam admission 50% of the achievable points for all homework exercises in total. For higher scores up to 2 bonus points for the exam.
- Questions and comments to: gkemnitz@in.tu-clausthal.de

1.4 Examination

- Written exam for 10 or more participants.
- Permitted aids for the written examination: own elaboration including handouts with own comments and own homework, calculator.
- Permitted aids for oral examination: One A4 sheet (single-sided) with own elaborations.

For all further information see website.

1.6 Trust and dependability



IT systems automate intellectual tasks:

- operational procedures,
- control of processes and machinery,
- design tasks, ...

A precondition for use is confidence that

- the system will work when it is needed,
- performs its services correctly and in time,
- does not cause incalculable damage and costs.

Trust in an IT system depends on dependability.

1.6 Dependability

In everyday language, dependability (of people, computers, ...) means that you can trust them. Various aspects come together here (wishes, expectations, ...).

Individual factors influencing the perception of dependability:

- Life experiences, especially from childhood,
- Catastrophes or slow changes,
- Personality type (optimist, pessimist, conservative, gambler), ...

Objectification by counting positive and negative experiences and descriptive attributes:

- for what reliable:
  - Lecturer reliable that on time,
  - Student reliable that exams are handed in, ...
- why reliable:
  - Doctor reliable because completed medical studies,
  - Car reliable because technical licence and valid MOT.

Passed tests are important, but also the error culture ...

1.7 Error-handling culture

The way in which societies, cultures and social systems deal with errors and their consequences.

Negative view: Hiding mistakes, talking them away, ...

Positive views: Learn from mistakes, eliminate mistakes. ...

- Pedagogy: positive climate for learning from mistakes.
- Quality management: minimising the cost of errors.
- Innovation management: striving for innovation. Errors as Opportunity / productive potential.

The lecture assumes an idealised error-handling culture:

- all identified problems are rectified and
- the rectification will be verified.

In human interactions with friends, superiors and partners and also in cost optimisation of design, production, ... less radical (more tolerant) error cultures are usually better.

---

* yA simplified model that neglects economic and cultural aspects.

1.10 Hazards and hazard prevention

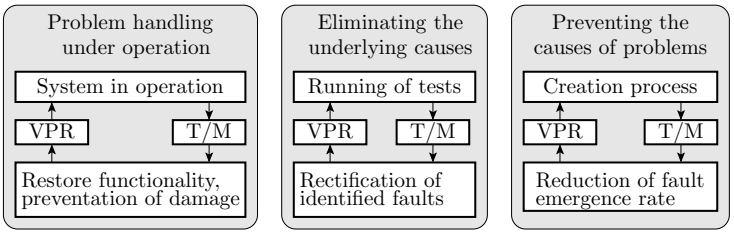
dependability is described by the absence / prevention / exclusion of hazards, on three levels [2]:

1. Problems during operation:
   - crash and other types of denial of service (no service, NS).
   - malfunction (MF),
2. Root causes of the problems of (1):
   - faults (eliminateable causes for NS and MF),
   - disturbances (causes of randomly occurring problems),
   - failures (faults occurring during operation).
3. Origins of the causes of problems:
   - shortcomings, faults,
   - disturbances and failures

   in the development and repair processes.

MF, NS yMalfunction. No service.

1.11 Levels of hazard avoidance



Hazard prevention in line with our idealised error culture, through a series of iterations comprising checks, attempts to resolve problems and success checks at three levels:

- Dealing with problems under use
- Testing and fault rectification before deployment.
- Fault prevention by improving creation processes.

T/M, VPR yTesting and monitoring for problems. Verification of Problem Resolution.

1.12 What is the cost of dependability?

Costs for hazard prevention:

- Monitoring and adequate response to detected malfunctions: up to more than 50 per cent of overall functionality, e.g. when doubling and comparing results.
- Testing, troubleshooting and debugging: For Hardware and Software typically more than 50% of total design cost.
- Fault prevention by improving the development processes: Costs for quality assurance and the ongoing development and improvement of the development processes.

Overall, measures to ensure dependability can account for well over 50 per cent of product costs.

dependability is the most expensive feature of an IT system.

1.13 The price of insufficient dependability

Since reliability is expensive, why not do without it? – Costs of damage:

- no or limited usability,
- data loss, backdoors for data misuse[1],
- Accidents, self-destruction, production downtime, ...

> On 3 June 1980, a computer at the North American Air Defence Centre reported the approach of Soviet nuclear missiles. Retaliatory measures were immediately prepared. A review of the data from radar stations and satellites could not confirm the attack ... [1].

Cause of the almost nuclear strike: a defective circuit.

Only reliable IT systems are ready for use.

1.14 Hypothetical insurance against damage

Spectacular disaster scenarios raise awareness of problems, but are not suitable for cost-benefit analyses.

The cost of a lack of dependability can best be estimated by a hypothetical insurance policy covering all potential against damage caused by IT using insurance mathematics.

Hypothetical, as no such insurance policies exist and, from a legal perspective, it is difficult to offer due to the historically established distribution of risk:

- Separate business risks for IT manufacturers, operators and users, for each stakeholder, in terms of financial loss, damage to reputation and loss of customers, ...
- product liability in the case of quality assurance shortcomings,
- major damage for which the public has to pay.

Reliability measures are cost-effective if the savings on insurance policies exceed the additional costs.

In the case of entertainment IT, for example, the acceptable costs are much lower than for IT in the aviation sector.

1.15 Naming the lecture

Dependability, as the most costly product characteristic of IT systems, is subject to many aspects, of which the **tests** and checks carried out are the most significant.

- **Dependability** is the umbrella term for availability, reliability and safety, and
- all three sub-aspects are ensured at all three levels – namely problem handling, troubleshooting and fault prevention – through checks (monitoring, testing, etc.).

In our idealised error-handling culture, all identified problems are rectified and the rectification is verified:

- only undetected problems cause damage that, hypothetically, would need to be insured against, and
- the frequency of undetected problems depends largely on the effectiveness of the checks and tests.

1.16 Learning objective and content

Learning objective

Basic understanding of what dependability is, of what measures it is influenced by and how. A subject-specific introduction to stochastics, methods of monitoring and testing, and key aspects and specific features of hardware and software.

Foliensätze:

[1] https://www.faz.net/aktuell/wirtschaft/diginomics/43-milliarden-euro-schaden-durch-hackerangriffe-15786660.html

1. Measurements. Problem handling during operation.
2. Test, Fault rectification. Fault prevention.
3. Stochastics I: Probabilities.
4. Stochastics II: Estimates and Forecasts.
5. More about tests and checks.
6. Hardware testing and self-test.
7. Software design and testing.

1 Dependability

1.17 Quantifying Dependability

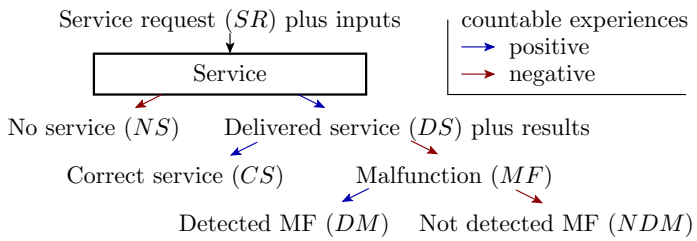
Usage of IT is based on trust. dependability describes the extent to trust them. Quantifiable through counts of positive and negative experiences across different aspects:

- Delivery: Positive: delivery on demand. Negative: no service.
- Correctness: Positive: correct results. Negative: malfunction.
- Hazzards: Positive: no safety risk. Negative: safety risks of varying types and severity.

For a quantitative assessment based on count data IT systems must be modelled in such a way that services delivered and not delivered, correct and incorrect results, and other potential and avoided problems under consideration are countable.

1.1 Service-Modell

1.18 Service provider



Basic model for quantifying positive and negative experiences. **System that generates outputs from inputs upon request**:

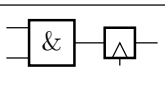
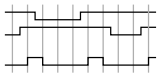
- Positive experience: delivered service (DS),
- Negative experience: no service (NS).

A service provided may be:

- Positive experience: correct service (CS),
- Negative experience: malfunction (MF).

1.19 Areas of application for the service model

The service model is applicable to different types and abstractions of IT systems, human and automated services, as well as manufacturing and design processes, ...

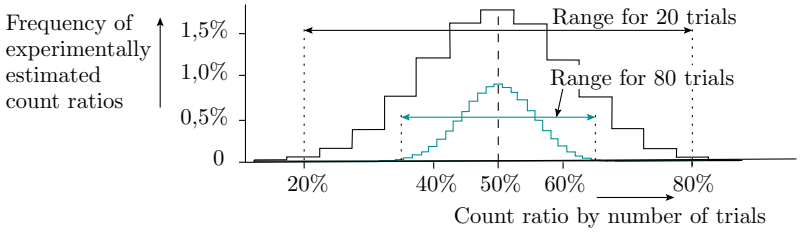
Clocked digital circuit		I: O: 

Programme with IPO-structure	`uint8_t up(uint8_t a){`
	`  return 23 * a;`
	`}`
		I: 10 101 ...
		O: 230 19 ...

Server	I: e.g. database enquiry O: Result dataset	
Manufacturing process	I: Production order, material, ... O: Manufactured product	
Design process	I: Design order O: Design	

I, P, O yInput. Processing. Output.

1.20 Sufficiently large count values (SLC)



Defining dependency parameters using proportionate count values for positive or negative outcomes allows for a simple and clear assessment of actual or simulated counting experiments:

```
for 10^5 repetitions:
  for n = 20 or 80 trials, respectively:
    if rand( ) < 50%:
      increase the count by 1
```

Result:

n	ε
20	30%
80	15%

n, ε yNumber of counting trials. Interval radius of the determined frequencies.

Minimum count value	$\varepsilon_{rel} = 10\%$	$\varepsilon_{rel} = 5\%$	$\varepsilon_{rel} = 2\%$
$\alpha = 4\%$	416	1665	10.404
$\alpha = 2\%$	543	2172	13.572
$\alpha = 1\%$	660	2642	16.512

The table shows the minimum count value as a function of the permissible relative interval radius ε_{rel} and the error probability α for independent counts and small occurrence frequencies (sec.4.2.7). The relative interval radius is the ratio of the interval radius to the occurrence frequency. The probability of error is the probability that, if the experiment were repeated, the result would lie outside the estimated interval.

Accurate estimates require large count values and therefore many counting trials. This will be covered in the topic in Slide Set 4. Until then, all parameters to be estimated via counting trials are labelled with $\ldots|_{SLC}$:

Reliable estimates only with sufficient counting trials.

ε_{rel}, α yRelative interval radius, error probability.

SLC yUseful estimates only with Sufficiently large count values.

1.2 Availability

1.22 Availability metrics

Definition of availability as the rate at which requested services are delivered:

$$A = \left.\frac{\#DS}{\#SR}\right|_{\text{SLC}} \tag{1.1}$$

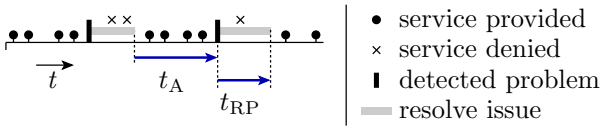
An alternative measure is the probability of failure on demand:

$$PFD = 1 - A \tag{1.2}$$

There are various reasons for unavailability, each of which requires a different reaction. Separate counting for partial availabilities that can be combined to obtain total availability (slide 1.27).

A	Availability.
$\#SR$	Number of service requests.
$\#DS$	yNumber of delivered services.
PFD	yProbability of failure on demand.
SLC	yUseful estimates only with Sufficiently large count values.
Rate	yRelative frequency of occurrence of the event under consideration.

1.23 Availability, time taken to resolve issues



A system always operates for a period of time t_A without any visible problems (i.e. without denial of service, crashes, detected malfunctions, etc.), followed by a problem-resolution time t_{PR} for repair, reinitialisation, etc. Availability as the average proportion of time:

$$A = \frac{\bar{t}_A}{\bar{t}_A + \bar{t}_{RP}} \tag{1.3}$$

Significant dependence on the average problem resolution time.

A	Availability.
t_A, $\bar{t}_A$	yTime and average time during which the system is available between issues.
t_{RP}, $\bar{t}_{RP}$	yTime and average time taken to resolve the issue.

1.24 Problem resolution – tasks, effort involved

An IT system should respond appropriately to service denial, system crashes and detected malfunctions:

- Mitigation of damage: data recovery, fallback to a safe state, ...
- Restoring operational readiness: repair, reinitialisation, ...
- Logging and storing data to identify the cause of the problem,
- Termination without service provision or a new request.

In particular, repairs following hardware failures, but also data recovery, ensuring secure conditions, and re-ordering may take a considerable amount of time, during which the system is unavailable.

Availability can be improved by reducing the frequency of problems and by shortening problem-resolution times.

1.25 Partial availabilities

Unavailability can have various causes:

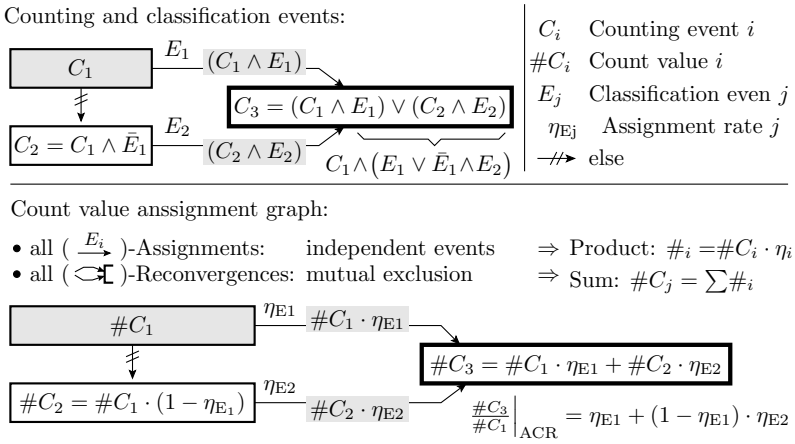
1. Hardware failure (FL)
2. Denial of acceptance (DA),
3. Crash (CR),
4. Detected malfunction (DM).

Each has its own count values, its own response measures, and its own occurrence and tolerance rates.

	Failure	DA	Crash	DM
Count of negative events	$\#FL$	$\#DA$	$\#CR$	$\#DM$
Count of positive events	$\#HA$	$\#RA$	$\#DR$	$\#DS$
Occurrence rate	ζ_{FL}	ζ_{DA}	ζ_{CR}	ζ_{DM}
Tolerance rate	ν_{FL}	ν_{DA}	ν_{CR}	ν_{DM}

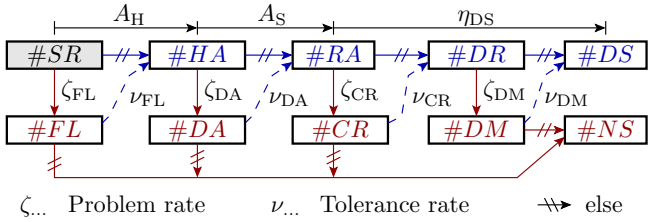
Introduction of a graphical representation of complex relationships between measured values, enabling a clear visualisation and easy interpretation of measured values and rates.

1.26 CVA (count-value assignment) graph



See »Chained Events« below, section 3.1.2.

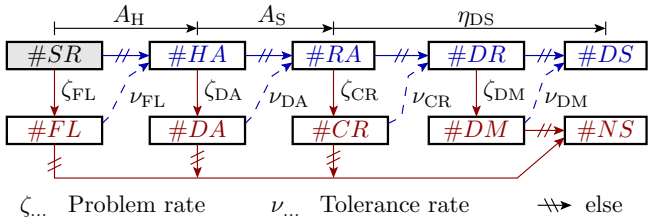
1.27 Breakdown into three partial availabilities



	Failure	DA	Crash	DM
Count of negative events	$\#FL$	$\#DA$	$\#CR$	$\#DM$
Count of positive events	$\#HA$	$\#RA$	$\#DR$	$\#DS$
Occurrence rate	ζ_{FL}	ζ_{DA}	ζ_{CR}	ζ_{DM}
Tolerance rate	ν_{FL}	ν_{DA}	ν_{CR}	ν_{DM}

- $\#HA$ counts requests for which hardware is available,
- $\#RA$ counts requests, for which the service is also available,
- $\#DR$ counts the accepted services provided, ...

Pronounce: yζ: zeta, ν: ny.



Hardware availability

$$A_H = \frac{\#HA}{\#SR}\Big|_{SLC}$$
$$A_H = (1-\zeta_{FL}) + \zeta_{FL}\cdot\nu_{FL} \tag{1.4}$$

Service availability:

$$A_S = \frac{\#RA}{\#HA}\Big|_{SLC}$$
$$A_S = (1-\zeta_{DA}) + \zeta_{DA}\cdot\nu_{DA} \tag{1.5}$$

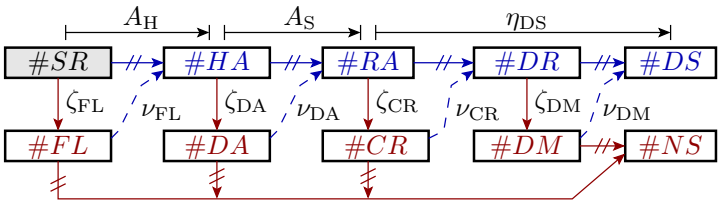
Delivery rate:

$$\eta_{DS} = \frac{\#DS}{\#RA}\Big|_{SLC} = \ldots$$

Overall availability:

$$A = \frac{\#DS}{\#SR}\Big|_{SLC}$$
$$A = A_H \cdot A_S \cdot \eta_{DS} \tag{1.6}$$

A, η_{DS} yAvailability. Rate of delivered services.
A_H, A_S yHardware availability. Service availability.
ζ_{FL}, ζ_{DA} yHW unavailability rate. Service denial rate.

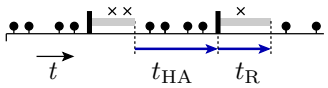


For independent events: failure, failure tolerance, ... along all paths to $\#HA$, then on to $\#RA$, ... only

- independent events are combined using »AND« and
- mutually exclusive events are combined using »OR«.

$\#\langle evt\rangle$ yNumber of count events:.
SR, HA yService request. Hardware available.
RA, DR yService Request accepted. Delivered result.
DS, NS yDelivered Service. No service.
FL, DA yNon-availability due to hardware failure or denial of acceptance.
CR, DM yUnavailability due to crash or recognised malfunction.
ζ_{FL}, ν_{FL} yHW unavailability rate. Tolerance rate for unavailable HW.
ζ_{DA}, ν_{DA} yService denial rate. Tolerance rate for service denials.
ζ_{CR}, ν_{CR} yCrash rate. Rate of tolerance of crashes.
ζ_{DM}, ν_{DM} yRate of detected malfunctions. Tolerance rate for detected malfunctions.

1.30 Hardware availability



$$A_H = (1 - \zeta_{FL}) + \zeta_{FL} \cdot \nu_{FL} \tag{1.4}$$

Also, the proportion of time the system is available (eq. 1.3):

$$A_H = \frac{\bar{t}_{HA}}{\bar{t}_{HA}+\bar{t}_R} \overset{\bar{t}_{HA} \gg \bar{t}_R}{=} 1 - \frac{\bar{t}_R}{\bar{t}_{HA}} \tag{1.7}$$

High availability requires a low rate of (total) failures, short repair times and, where possible, redundancies that take over the tasks of failed components (sec.6.5.2). Total failure as opposed to minor failures, after which the system remains usable, but with reduced reliability (see section 6.5.2 below).

A_H yHardware availability.

ζ_{FL}, ν_{FL} yHW unavailability rate. Tolerance rate for unavailable HW.

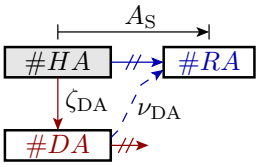
t_{HA}, $\bar{t}_{HA}$ yService life and average service life between total hardware failures.

t_R, $\bar{t}_R$ yRepair time. Average repair time.

Total failure: Failure after which a component is no longer usable and must be repaired.

Minor failure: Failure-induced fault that reduces only reliability.

1.31 Service availability



$$A_S = (1 - \zeta_{DA}) + \zeta_{DA} \cdot \nu_{DA} \tag{1.5}$$

Service refusal when hardware is available, e.g. due to:

- Troubleshooting following a crash or a detected malfunction,
- Exceeding the permitted processing time,
- Overload due to too many service requests, ...

Can be avoided by ensuring sufficient capacity. Can be tolerated by allowing for delays in processing, redistributing tasks, etc.

Closely intertwined with functional aspects. Further on, it is usually neglected ($A_S = 1$).

A_S yService availability.

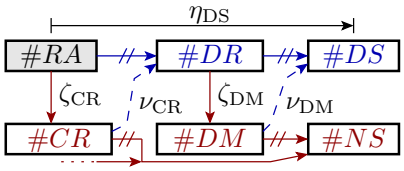
$\#HA$ yNumber of requests to available hardware.

$\#RA$ yNumber of accepted service requests.

$\#DA$ yNumber of denials of acceptance.

ζ_{DA}, ν_{DA} yService denial rate. Tolerance rate for service denials.

1.32 Delivery of accepted requirements



Service delivery rate for accepted requirements

$$\eta_{DS} = \left.\frac{\#DS}{\#RA}\right|_{SLC}$$

depends on

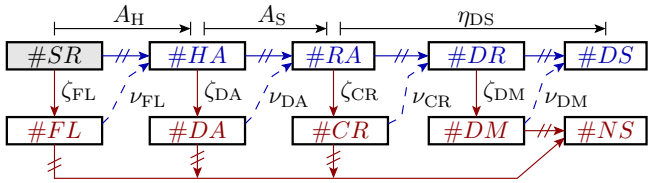
- the crash rate ζ_{CR}, the rate of detected malfunctions ζ_{DM} ,

- the associated tolerance rates ν_{CR} and ν_{DM}, and
- how detected malfunctions are handled.

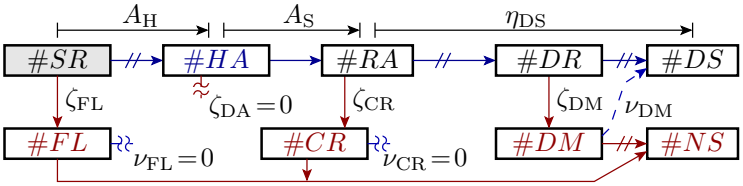
Continued in section 1.2: Troubleshooting.

RA, *DR*	yService Request accepted. Delivered result.
DS, *NS*	yDelivered Service. No service.
CR, *DM*	yUnavailability due to crash or recognised malfunction.

Example 1.1 CVA graph and availability



a) *Adjust the CVA graph so that failures are never tolerated, service requests are always accepted when hardware is available, and no service is provided in the event of a crash.*



b) *What are the formulas for hardware availability, service availability, service delivery rate and overall availability, as shown in the CVA graph from part (a) of the question?*

$$A_{\mathrm{H}} = (1-\zeta_{\mathrm{FL}})\,;\; A_{\mathrm{S}} = 1$$
$$\eta_{\mathrm{DS}} = (1-\zeta_{\mathrm{CR}})\cdot(1-\zeta_{\mathrm{DM}}+\zeta_{\mathrm{DM}}\cdot\nu_{\mathrm{DM}})$$
$$A = (1-\zeta_{\mathrm{FL}})\cdot(1-\zeta_{\mathrm{CR}})\cdot(1-\zeta_{\mathrm{DM}}\cdot(1-\nu_{\mathrm{DM}}))$$

SR, *HA*	yService request. Hardware available.
RA, *DR*	yService Request accepted. Delivered result.
DS, *NS*	yDelivered Service. No service.
FL, *DA*	yNon-availability due to hardware failure or denial of acceptance.
CR, *DM*	yUnavailability due to crash or recognised malfunction.
ζ_{FL}, ν_{FL}	yHW unavailability rate. Tolerance rate for unavailable HW.
ζ_{DA}, ν_{DA}	yService denial rate. Tolerance rate for service denials.
ζ_{CR}, ν_{CR}	yCrash rate. Rate of tolerance of crashes.
ζ_{DM}, ν_{DM}	yRate of detected malfunctions. Tolerance rate for detected malfunctions.
A_{H}, A_{S}	yHardware availability. Service availability.
η_{DS}, A	yRate of delivered services, Overall availability.

1.3 Reliability

1.34 A Look back

The use of IT requires trust. Dependability describes the extent to which this trust is justified. An objective description based on metrics for positive and negative experiences. Classification by aspects:

Service delivery:

- Positive experiences: Service provided upon request.
- Negative experiences: Refusal of service.
- Performance indicators: Availability, which can be broken down into the product of partial availabilities.

Accuracy:

- Positive experiences: the results produced were correct.
- Negative experiences: the results produced were incorrect.
- Key figures: <continue here>

Damage:

- Positive experiences: no damage when requesting a service.
- Negative experiences: damage of various types and degrees.
- Key figures: (sec. 1.1.4)

1.35 Key Figures for Accuracy

Reliability*: Number of delivered services (DS) per delivered malfunction (DMF):

$$R = \left.\frac{\#DS}{\#DMF}\right|_{\text{SLC}} \tag{1.8}$$

Malfunction rate: The reciprocal of reliability:

$$\zeta = \frac{1}{R} = \left.\frac{\#DMF}{\#DS}\right|_{\text{SLC}} \tag{1.9}$$

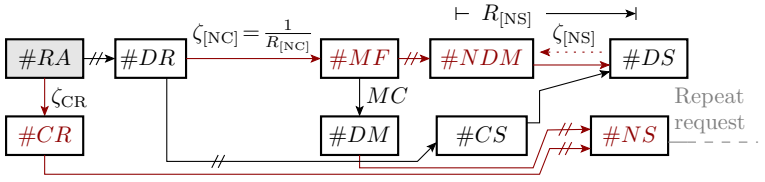
Probability of reliability as the counter-probability of the Malfunction rate:

$$p_{\text{R}} = 1 - \zeta$$

The preferred reliability measure is R , because in this case a tenth of the failures equates to ten times the reliability $10 \cdot R$, rather than an increase in the probability of reliability from 99% to 99.9%.

R, ζ	yReliability. Malfunction rate.
$\#DS$	yNumber of delivered services.
$\#DMF$	yNumber of malfunctions delivered.
SLC	yUseful estimates only with Sufficiently large count values.
*	yPurposeful but still uncommon definitions in the professional world.

1.36 Update of the CVA Graph



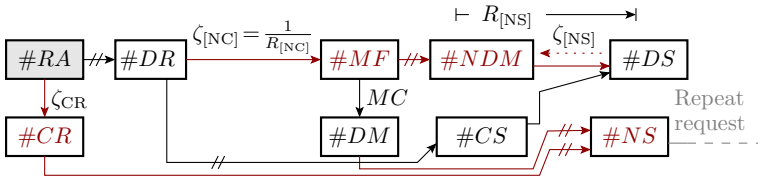
Adding count values for undetected faults: Malfunctions (MF) are detected with frequency MC become (DM) and otherwise remain undetected (NDM).

In the example CVA graph, no service delivery (NS) in the event of crash (CR) or detected malfunction (DM).

Later, we will see, this also applies with at most one re-request [SR] and re-requests until service delivery [RR] (sec.1.2.5).

$\#\langle evt\rangle$	yNumber of count events:.
RA, DR	yService Request accepted. Delivered result.
MF, NDM	yMalfunction. Undetectable malfunction.
DS, NS	yDelivered Service. No service.

1.37 Delivery Rate



Delivery rate of services in case of problems:

$$\eta_{DS} = (1-\zeta_{CR})\cdot\left(1-\zeta_{[NS]}\cdot MC\right) \tag{1.10}$$

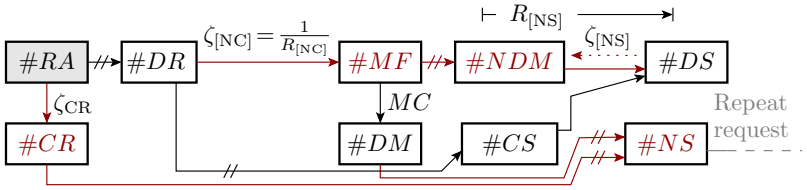
With the rate of observable malfunctions:

$$\zeta_{OMF} = \zeta_{[NC]}\cdot MC \tag{1.11}$$

$$\eta_{DS} = (1-\zeta_{CR})\cdot(1-\zeta_{OMF}) \tag{1.12}$$

$\zeta_{[NC/NS]}$	yMalfunction rate without checks / with no service for identified problems.
$R_{[NC/NS]}$	yReliability without monitoring / when skipping in case of detected problems.
ζ_{CR}, MC	yCrash rate. Malfunction coverage.
η_{DS}	yRate of delivered services.
ζ_{OMF}	yRate of observable malfunctions.

1.38 Reliability Improvement by Monitoring



Malfunction rate with monitoring and skipping bad outcomes:

$$\zeta_{DMF} = \frac{1}{R} = \frac{\#NDM}{\#DS}\Big|_{SLC} = \frac{\cancel{\#DR}\cdot(1-\zeta\cdot MC)}{\cancel{\#DR}\cdot\zeta\cdot(1-MC)}$$

$$\zeta_{DMF} = \frac{\zeta\cdot(1-MC)}{1-\zeta_{OMF}} \tag{1.13}$$

Reliability improvement:

$$R_{[NS]} = \frac{R_{[NC]}\cdot(1-\zeta_{OMF})}{(1-MC)} \tag{1.14}$$

ζ_{OMF}	yRate of observable malfunctions.
$R_{[NC/NS]}$	yReliability without monitoring / when skipping in case of detected problems.
MC, MT	yMalfunction coverage. Malfunction permeability.

The rate of observable malfunctions is generally very low ($\zeta_{\text{OMF}} \ll 1$). Reliability Improved through monitoring and skipping problems with this simplification:

$$R_{[\text{NS}]} = \frac{R_{[\text{NC}]}}{1-MC} = \frac{R_{[\text{NC}]}}{MT} \tag{1.15}$$

The reduction of the delivery rate due to skipping bad outcomes

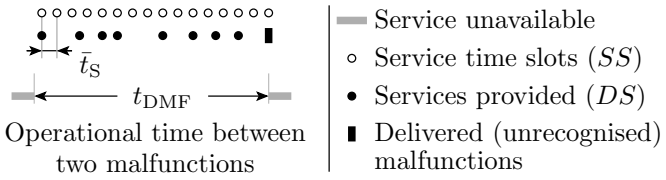
(1.12) $$\eta_{\text{DS}} = (1-\zeta_{\text{CR}}) \cdot (1-\zeta_{\text{OMF}})$$

is generally negligible and can be improved by repeated service requests.

Confirmation of the hypothesis (slide 1.15): Reliability is determined primarily by the quality of the checks carried out.

ζ_{OMF} yRate of observable malfunctions.
$R_{[\text{NC/NS}]}$ yReliability without monitoring / when skipping in case of detected problems.
MC, MT yMalfunction coverage. Malfunction permeability.
η_{DS} yRate of delivered services.
ζ_{OMF} yRate of observable malfunctions.
ζ_{CR} yCrash rate.

1.40 Estimation from Average Times



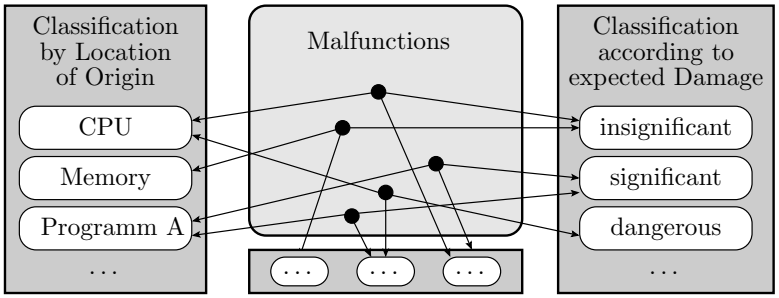
(1.8) $$R = \left.\frac{\#DS}{\#DMF}\right|_{\text{SLC}}$$

Estimation of reliability based on the mean time between delivery of undetected malfunctions and the mean duration of service:

$$R = \frac{\eta_{\text{SS}} \cdot \#SS}{\#DMF} \quad \text{with} \quad \frac{\#SS}{\#DMF} = \frac{\bar{t}_{\text{DMF}}}{\bar{t}_{\text{S}}}$$
$$R = \frac{\eta_{\text{SS}} \cdot \bar{t}_{\text{DMF}}}{\bar{t}_{\text{S}}} \tag{1.16}$$

$\#SS$, η_{SS} yNumber of service time slots per delivered malfunction.
$\#DMF$ yNumber of malfunctions delivered.
$\bar{t}_{\text{DMF}}$, $\bar{t}_{\text{S}}$ yMean service time between unrecognised malfunctions. Mean service time.
R Reliability.

1.41 Partial Reliabilities



The problems within a system can also be counted separately according to different criteria, e.g.

- by location, cause, damage,
- separately for subsystems (CPU, memory, ...),

- separated into faults, disruptions and failures,
- only issues that compromise operational safety or data security.

If each malfunction is assigned to exactly one class* , the sum of all malfunctions is equal to the sum of the malfunctions of all classes:

$$\#MF = \sum_{i=1}^{\#MFC} \#MF_i$$

Total malfunction rate equal to the sum of all partial malfunction rates:

$$\zeta = \sum_{i=1}^{\#MFC} \zeta_i \tag{1.17}$$

The reciprocal of the overall reliability is equal to the sum of the reciprocals of the partial reliabilities:

$$\frac{1}{R} = \sum_{i=1}^{\#MFC} \frac{1}{R_i} \tag{1.18}$$

This applies regardless of how malfunctions are handled (without monitoring, with skipping or new request in case of problems, ...) and also holds for detected, avoided, ... malfunctions, for crashes, ...

$\#MFC$ yNumber of malfunction classes.
MF, MF_i yMalfunction. malfunction of class i.
ζ, ζ_i yTotal failure rate. Partial failure rate i.
R, R_i yOverall reliability. Partial reliability i.

Example 1.2 Partial reliabilities

In a system, malfunctions are caused by the memory, the processor, the software or other components. Mean time between detected (unrecognised) malfunctions in the subsystems:

Subsystem i	Memory	Processor	Software	others
$\bar{t}_{\mathrm{DMF}.i}$	1,000 h	6,000 h	2,000 h	4,000 h

Average service duration $\bar{t}_S = 1\,\mathrm{min}$. System utilisation $\eta_{SS} = 50\%$.

a) *What are the values of the four partial failure rates ζ_i and partial reliabilities that can be estimated from the time data? R_i?*

(1.16) $R = \frac{\eta_{SS} \cdot \bar{t}_{DMF}}{\bar{t}_S}$

Subsystem i	Memory	Processor	Software	others
$\eta_{SS} \cdot \bar{t}_{DMF}$ in min	$3 \cdot 10^4$	$18 \cdot 10^4$	$6 \cdot 10^4$	$12 \cdot 10^4$
R_i in $\left[\frac{DS}{DMF}\right]$	$3 \cdot 10^4$	$18 \cdot 10^4$	$6 \cdot 10^4$	$12 \cdot 10^4$
$\zeta_i = \frac{1}{R_i}$ in $\left[\frac{DMF}{DS}\right]$	$3{,}33 \cdot 10^{-5}$	$5{,}56 \cdot 10^{-6}$	$1{,}67 \cdot 10^{-5}$	$8{,}33 \cdot 10^{-6}$

b) *What are the failure rate ζ and reliability R of the overall system?*

(1.9) $\zeta = \frac{1}{R} = \left.\frac{\#DMF}{\#DS}\right|_{SLC}$

(1.17) $\zeta = \sum_{i=1}^{\#MFC} \zeta_i$

(1.18) $\frac{1}{R} = \sum_{i=1}^{\#MFC} \frac{1}{R_i}$

$$\begin{aligned}
\zeta &= \left(3{,}33 \cdot 10^{-5} + 5{,}56 \cdot 10^{-6} + 1{,}67 \cdot 1 + 8{,}33 \cdot 10^{-6}\right) \left[\tfrac{DMF}{DS}\right] \\
&= 6{,}39 \cdot 10^{-5} \left[\tfrac{DMF}{DS}\right] \\
\frac{1}{R} &= \left(\tfrac{1}{3 \cdot 10^4} + \tfrac{1}{18 \cdot 10^4} + \tfrac{1}{6 \cdot 10^4} + \tfrac{1}{12 \cdot 10^4}\right) \left[\tfrac{DMF}{DS}\right] \\
R &= \tfrac{1}{\zeta} = 1{,}57 \cdot 10^4 \left[\tfrac{DS}{NDM}\right]
\end{aligned}$$

$\bar{t}_{DMF.i}$ yMean service time between delivery of malfunctions of subsystem i.
$\bar{t}_S$, η_{SS} yAverage service duration. System utilisation rate.
ζ, ζ_i yTotal failure rate. Partial failure rate i.
R, R_i yOverall reliability. Partial reliability i.
R_i, ζ_i yPartial reliability i. Malfunction rate of subsystem i.
$[\frac{DS}{DMF}]$ yCounting value ratio in elivered services per malfunction.
$[\frac{DMF}{DS}]$ yCounting value ratio in malfunctions per delivered service.

1.4 Safety (security)

1.44 Damage caused by IT

Safety for industrial equipment is currently described in terms of safety levels. SIL (**S**afety **I**ntegrity **L**evel) in accordance with IEC 61508:

- SIL1: Minor damage to facilities and property,
- SIL2: Serious damage to equipment, harm to persons,
- SIL3: Injuries, some fatalities,
- SIL4: Disasters, many deaths, serious environmental damage.

The SIL between others set key performance indicators for availability:

- PFH (**p**robability of **f**ailure per **h**our),
- PFD (**p**robability of **f**ailure on **d**emand), ...

SIL	1	2	3	4
$PFH_{\max}$	10^{-5}	10^{-6}	10^{-7}	10^{-8}
$PFD_{\max}$	10^{-1}	10^{-2}	10^{-3}	10^{-4}

Quantifying availability using target metrics does not seem to be the right approach. We will define **safety** as the **number of service requests** (SR) **per safety-related problem** (SP) and a subdivision types of risks.

1.45 Safety-related Issues

Safety (security) refers to assumed risks:

Safety (security)	Safe from what?
Operational safety	Injury, environmental damage
Access security (data protection)	Data theft
Data security	Data loss
...	...

Rate of problems posing a safety risk / compromise security:

$$\zeta_S = \left.\frac{\#SP}{\#SR}\right|_{SLC} \tag{1.19}$$

If each problem is assigned to exactly one problem class, the overall rate is the sum of the sub-problem rates of all problem classes:

$$\zeta_S = \sum_{i=1}^{\#SPC} \zeta_{S.i} \tag{1.20}$$

SR, SP yServices requests. Safety-related problem.
$\#SPC$ yNumber of safety and security problem classes.
ζ_S, $\zeta_{S.i}$ yRate of safety endangering malfunctions in total, for each problem class individually.

1.45 Overall and partial Safety

Safety is the reciprocal of the rate of safety-compromising problems:

$$S = \left.\frac{\#SR}{\#SP}\right|_{\text{SLC}}$$
$$S = \frac{1}{\zeta_{\text{S}}} \tag{1.21}$$

If each problem is assigned to exactly one problem class, the reciprocal of the overall safety is equal to the reciprocal of the sum of all partial safeties:

$$\frac{1}{S} = \sum_{i=1}^{\#SPC} \frac{1}{S_i} \tag{1.22}$$

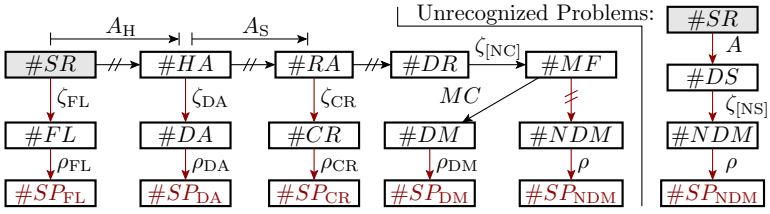
All problems in an IT system

- Hardware failure, denial of service,
- System crash, detected/undetected malfunction.

may pose a safety (security) risk at a rate of $\rho_{[..]}$.

S, S_i	yOverall safety. Partial safety i.
$\#SPC$	yNumber of safety and security problem classes.
SLC	yUseful estimates only with Sufficiently large count values.

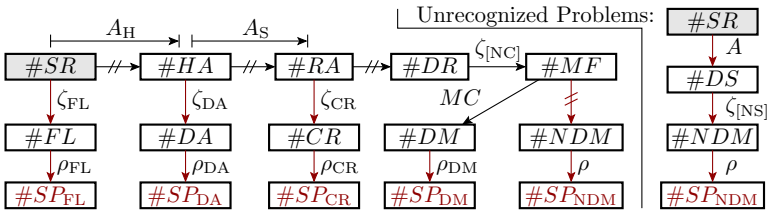
1.46 CVA Graph of Safety (Security) Hazards



Rates of safety hazards caused by hardware failures:

$$\zeta_{\text{S.FL}} = \zeta_{\text{FL}} \cdot \rho_{\text{FL}} \tag{1.23}$$

SR, FL	yService request. Hardware failed.
HA, DA	yHardware available. Acceptance denied.
RA, CR	yRequest accepted. Crash.
DS, MF	yDeliverd services. Malfunctions.
ζ_{FL}, ζ_{DA}	yHW unavailability rate. Service denial rate.
ζ_{CR}, MC	yCrash rate. Malfunction coverage.
$\zeta_{[\text{NC/NS}]}$	yMalfunction rate without checks / with no service for identified problems.
$\rho_{[..]}$, $\#SP_{[..]}$	yProportion and count value of safety-related problems.
Pronounce:	yζ: zeta, ρ: rho.



Rates of security risks arising from refusal to accept delivery:

$$\zeta_{\mathrm{S.DA}} = A_{\mathrm{H}} \cdot \zeta_{\mathrm{DA}} \cdot \rho_{\mathrm{DA}}$$

Incidence rates of safety hazards caused by crashes (CR):

$$\zeta_{\mathrm{S.CR}} = A_{\mathrm{H}} \cdot A_{\mathrm{S}} \cdot \zeta_{\mathrm{CR}} \cdot \rho_{\mathrm{CR}}$$

Rates of safety risks caused by detected malfunctions:

$$\zeta_{\mathrm{S.DM}} = A_{\mathrm{H}} \cdot A_{\mathrm{S}} \cdot (1 - \zeta_{\mathrm{CR}}) \cdot \zeta_{[\mathrm{NC}]} \cdot MC \cdot \rho_{\mathrm{DM}}$$

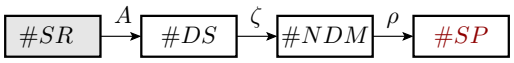
Rates of safety risks caused by undetected malfunctions:

$$\zeta_{\mathrm{S.NDM}} = A_{\mathrm{H}} \cdot A_{\mathrm{S}} \cdot (1 - \zeta_{\mathrm{CR}}) \cdot \zeta_{[\mathrm{NC}]} \cdot (1 - MC) \cdot \rho = A \cdot \zeta_{[\mathrm{NS}]} \cdot \rho$$

Overall rate of all problems posing a risk:

$$\zeta_{\mathrm{S}} = \underbrace{\zeta_{\mathrm{S.DA}} + \zeta_{\mathrm{S.CR}} + \zeta_{\mathrm{S.DM}}}_{\text{Risk prevention possible}} + \underbrace{A \cdot \zeta_{[\mathrm{NS}]} \cdot \rho}_{\text{Risk prevention impossible}}$$

1.49 Safe State



Ideally, the system switches to a safe state as soon as a problem is detected. Security risks are then limited to those arising from undetected malfunctions:

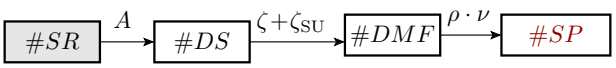
$$\zeta_{\mathrm{S}} = A \cdot \zeta \cdot \rho$$

For availability $A \approx 1$ and no safety risk arising from detected problems, safety is the quotient of reliability and the proportion of malfunctions that pose a safety risk:

$$S = \frac{R}{\rho} \tag{1.24}$$

SR, *DS*	yService request. Delivered service.
DS, *NDM*	yUnrecognised malfunction. Safety-critical issue.
A, *Z*, *S*	yAvailability. Reliability. Safety or security.
ζ, ρ	yMalfunction rate. Proportion of malfunctions that pose a safety risk.

1.50 Safety Improvment by a Safety Unit



Reduction of the safety-critical proportion ρ by means of an additional safety unit:

- Emergency power supply, emergency programmes,
- Protective devices, safety devices, ...

which reduce the proportion of safety-critical problems by a factor ν, but which themselves contributes to the number of undetected malfunctions by a rate of $\zeta_{\mathrm{SU}}{}^{*}$:

$$S_{\mathrm{ASU}} = \frac{1}{(\zeta + \zeta_{\mathrm{SU}}) \cdot \rho \cdot \nu} \tag{1.25}$$

SR, *DS*	yService request. Delivered service.
DS, *NDM*	yUnrecognised malfunction. Safety-critical issue.
ζ, ζ_{SU}	yRate of delivery malfunctions: system / safety unit.
ρ, ν	yShare of safety-critical malfunctions. Reduction of this share by the safetyty unit.
S_{ASU}	yOverall safety (security) of a system with an additional safety unit.
*	Availability one and switching to a safe state whenever a problem is detected.

Example 1.3 Improving Vehicle Safety

A vehicle has a mean time between undetected malfunctions of 1000 h. *The proportion of these that compromise operational safety is* 1%, *and the mean service time (mean driving time) is* 1 h. *System utilisation is one. An additional electronic control unit with reliability* R_{SU} *reduces the proportion of problems that compromise safety to one-tenth. Safety is compromised only by undetected malfunctions, availability one.*

$\bar{t}_{\mathrm{NDM}}$ =1000 $\left[\frac{\mathrm{h}}{\mathrm{NDM}}\right]$, $\bar{t}_{\mathrm{S}} = 1\ \left[\frac{\mathrm{h}}{\mathrm{DS}}\right]$, $\eta_{\mathrm{SS}} = 1$, $\rho = 1\%\ \left[\frac{\mathrm{SP}}{\mathrm{NDM}}\right]$, $\nu = 0{,}1$.

a) *Reliability and safety without the additional control unit?*

(1.16) $$R = \frac{\eta_{\mathrm{SS}} \cdot \bar{t}_{\mathrm{DMF}}}{\bar{t}_{\mathrm{S}}}$$

(1.24) $$S = \frac{R}{\rho}$$

Reliability:

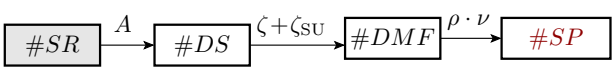
$$R = \frac{\eta_{\mathrm{SS}} \cdot \bar{t}_{\mathrm{NDM}}}{\bar{t}_{\mathrm{S}}} = 1000\ \left[\frac{\mathrm{DS}}{\mathrm{NDM}}\right]$$

Of which $\sigma = 1\%$ poses a safety risk:

$$S = \frac{R}{\rho} = 10^5\ \left[\frac{\mathrm{DR}}{\mathrm{SP}}\right]$$

b) *Minimum reliability* R_{SU} *of the auxiliary control unit, such that safety is increased at least fivefold* ($S_{\mathrm{ASU}} \geq 5 \cdot S$)*?*

(1.25) $$S_{\mathrm{ASU}} = \frac{1}{(\zeta + \zeta_{\mathrm{SU}}) \cdot \rho \cdot \nu}$$



Maximum reduction in reliability caused by the control unit:

$$S_{\mathrm{ASU}} = \frac{1}{(\zeta + \zeta_{\mathrm{SU}}) \cdot \rho \cdot \nu} \geq 5 \cdot S = \frac{1}{\zeta \cdot \rho}$$

$$(\zeta + \zeta_{\mathrm{SU}}) \cdot \nu = 0{,}1 \cdot (\zeta + \zeta_{\mathrm{SU}}) \leq \frac{\zeta}{5}$$

$$\zeta_{\mathrm{SU}} \leq \zeta$$

$$R_{\mathrm{SU}} \geq R = 10^3\ \left[\frac{\mathrm{DS}}{\mathrm{NDM}}\right]$$

The auxiliary control unit must be at least as reliable as the vehicle.

$\bar{t}_{\mathrm{NDM}}$, $\bar{t}_{\mathrm{S}}$	yMean service time between unrecognised malfunctions. Mean service time.
η_{SS}	ySystem utilisation rate, percentage of available service time used.
ρ, ν	yShare of safety-critical malfunctions. Reduction of this share by the safetyty unit.
S, S_{ASU}	ySafety without/with additional unit for safety improvement.
R, R_{SU}	yReliability of the system. Reliability of the safety unit.

1.52 Notes on the Exercise

There are currently ethical debates as to whether autonomous vehicles should run over children, pensioners, ... in critical driving situations. This is not a constructive approach:

- Smart responses, such as detecting and deciding who should be run over, require complex additional systems that tend to be unreliable.
- Unreliable safety devices are not a good idea.
- It is not possible to programme any emergency response at all for undetected malfunctions..

High levels of safety (security) require:

- high system reliability, high fault coverage,
- Safety measures and equipment designed to prevent safety risks arising from identified problems.
- Insurance for the system to cover the residual risk arising from unforeseeable – and therefore untreatable – problems.

Summary

1.53 Lecture: Test and Dependability

This lecture aims to reorganise the field of study that deals with the most costly feature of IT systems – dependability – for the purpose of presenting knowledge in a concise manner:

- building blocks of knowledge that build on one another,
- concise definitions and models,
- which cover many of the issues important in practice.

(𝕃) Important for the exam are the simple fundamental concepts, and the conditions under which they apply. After all, you will come across these in practice.

(𝔹) More complex examples are often used merely for illustrative purposes. In the exam, it is sufficient to correctly identify the underlying concepts when presented with slides or images, and to apply the formulas correctly when working through examples.

1.54 Key Points (𝕃)

Dependability describes the extent to which trust in an IT system is justified by counting positive and negative experiences: malfunctions, crashes, correct performance, ...

Estimates are reliable only if the count values are large enough.

Breaking down into countable experiences via the service model:

> A system that generates output from input on request.

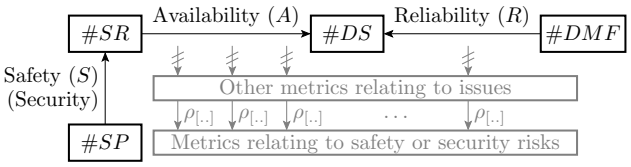
Dependability is ensured at three levels:

1. Dealing with problems during service,
2. Troubleshooting before and during use,
3. Fault prevention by means of improved design and manufacturing processes.

Each an iteration of checks, attempts to resolve problems and success checks. An idealised error culture in which all identified problems are resolved.

Definition of parameters using count-value assignment graphs.

1.55 Measures of Dependability (𝕃)



Measures for delivery:

- Availability: (1.1) $A = \frac{\#DS}{\#SR}\Big|_{\text{SLC}}$

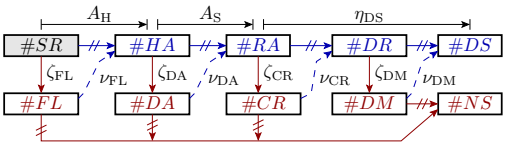
Measures of correctness:

- Reliability: (1.8) $R = \frac{\#DS}{\#DMF}\Big|_{\text{SLC}}$
- MF rate: (1.9) $\zeta = \frac{1}{R} = \frac{\#DMF}{\#DS}\Big|_{\text{SLC}}$

Characteristics of safety (security) risks:

- Rate safety probl.: (1.19) $\zeta_{\text{S}} = \frac{\#SP}{\#SR}\Big|_{\text{SLC}}$
- Safety (security): (1.21) $S = \frac{1}{\zeta_{\text{S}}}$

1.56 Overall and Partial Availability (𝔹*)



Categorised by cause of unavailability (and reaction to it):

- FL: Hardware failure,
- DA: Denial of acceptance,
- CR: Crash whilst running a service,
- DM: detected malfunction.

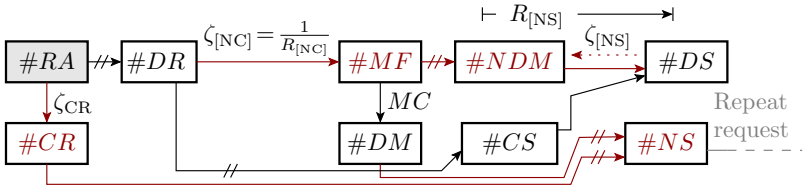
Hardware availability: (1.4) $A_{\text{H}} = (1 - \zeta_{\text{FL}}) + \zeta_{\text{FL}} \cdot \nu_{\text{FL}}$

Service availability: (1.5) $A_{\text{S}} = (1 - \zeta_{\text{DA}}) + \zeta_{\text{DA}} \cdot \nu_{\text{DA}}$

Overall availability: (1.6) $A = A_{\text{H}} \cdot A_{\text{S}} \cdot \eta_{\text{DS}}$

* Understanding the CVA graph. Derivation and application of the formulae.

1.57 Delivery Rate and Reliability



(𝔹) Delivery rate (without re-requests):

(1.10) $$\eta_{\mathrm{DS}} = (1 - \zeta_{\mathrm{CR}}) \cdot (1 - \zeta_{[\mathrm{NS}]} \cdot MC)$$

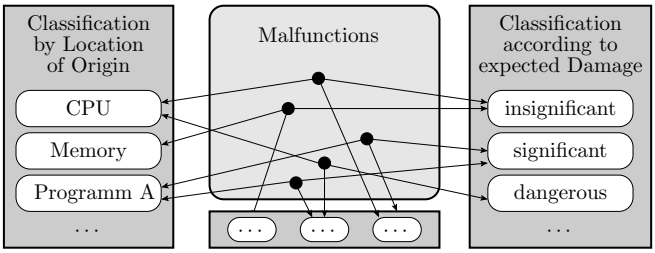
(𝔹) Reliability:

(1.14) $$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]} \cdot (1 - \zeta_{\mathrm{OMF}})}{(1 - MC)}$$

(𝕃) Where the rate of observable malfunctions is low $\zeta_{\mathrm{OMF}} \ll 1$:

(1.15) $$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]}}{1 - MC} = \frac{R_{[\mathrm{NC}]}}{MP}$$

1.58 Partial Reliabilities (𝕃)



If all malfunctions are assigned to classes in such a way that each one appears exactly once:

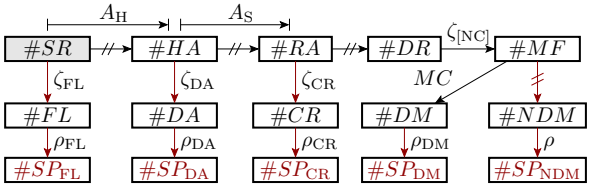
- Total failure rate equals the sum of the partial failure rates:

  (1.17) $$\zeta = \sum_{i=1}^{\#MFC} \zeta_i$$

- The reciprocal of the overall reliability is equal to the reciprocal of the sum of the partial reliabilities:

  (1.18) $$\frac{1}{R} = \sum_{i=1}^{\#MFC} \frac{1}{R_i}$$

1.59 Safety (Security) (𝔹)



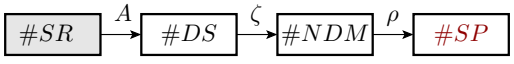
(𝕃) Classification of safety (security) measures by type of risk into operational, access and data safety/security.

(𝔹) Classification by cause and risk management:

- Unavailability due to failure, refusal, crashes,
- Malfunction functions, detected and undetected.

All these problems can lead to safety-critical issues with a low rate ρ. Partial and overall safety levels can be read from the CVA graph. Furthermore, it is assumed that a safe state is established whenever a problem is detected, so that the only remaining risks stem from undetected malfunctions.

1.60 Safe State, Safety Equipment (𝕃)



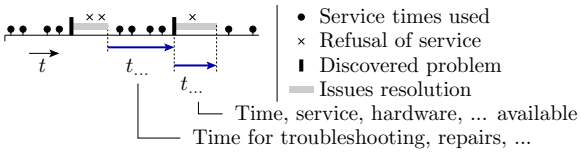
When transitioning to a safe state in the event of any detected problem (elimination of the risk of failures, crashes, detected malfunctions, etc.), **safety is proportional to reliability**:

(1.24) $$S = \frac{R}{\rho}$$

Safety equipment designed to reduce the proportion of safety-critical issues is itself a source of problems that compromises reliability. Overall safety improvement:

(1.25) $$S_{\mathrm{ASU}} = \frac{1}{(\zeta+\zeta_{\mathrm{SU}})\cdot\rho\cdot\nu}$$

1.61 Estimation Based on Time Measures (𝔹)



Availability in terms of mean time between problems and mean time to resolution:

(1.3) $$A = \frac{\bar{t}_{\mathrm{A}}}{\bar{t}_{\mathrm{A}}+\bar{t}_{\mathrm{RP}}}$$

Hardware availability as a function of the mean time between failures and the mean time to repair:

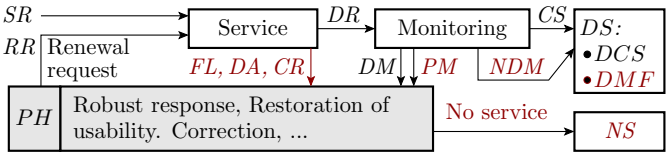
(1.7) $$A_{\mathrm{H}} = \frac{\bar{t}_{\mathrm{HA}}}{\bar{t}_{\mathrm{HA}}+\bar{t}_{\mathrm{R}}} \overset{\bar{t}_{\mathrm{HA}}\gg\bar{t}_{\mathrm{R}}}{=} 1-\frac{\bar{t}_{\mathrm{R}}}{\bar{t}_{\mathrm{FL}}}$$

Reliability as a function of the mean time between malfunctions and average service duration, etc.:

(1.16) $$R = \frac{\eta_{\mathrm{SS}}\cdot\bar{t}_{\mathrm{DMF}}}{\bar{t}_{\mathrm{S}}}$$

2 Problems under Use

1.62 Dealing with Problems under Use



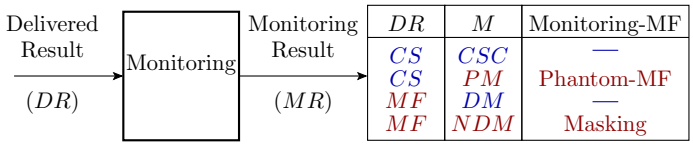
Monitoring for observable issues. If one detected:

- Establishing a safe system state.
- Notifying the user, optional sending an error report to the manufacturer.
- Restoring functionality (repair, reinitialization).
- Skipping the service or resubmitting the request.

SR, RR yService request. Repeat request.
DR, CS yDelivered result. Correct service.
NDM, PM yNot detected malfunction. Phantom malfunction.
DS, NS yDelivered service. No service.
FL, DA yNon-availability due to hardware failure or denial of acceptance.
CR, TS yCrash. Troubleshooting.
DMF yDelivered malfunction (faulty delivered service).

2.1 Monitoring

1.63 Monitoring



DR	M	Monitoring-MF
CS	CSC	—
CS	PM	Phantom-MF
MF	DM	—
MF	NDM	Masking

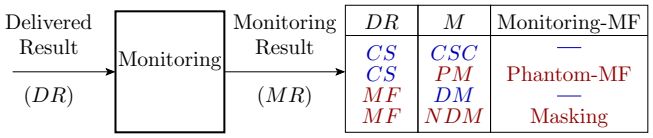
Responding to problems that arise during use (service denial, crashes, malfunctions, etc.) requires monitoring. In the simplest case, a monitoring system provides a binary 'good/bad' result, with the possible classification errors:

- Masking a detectable malfunction and
- Phantom malfunction.

Service refusals, e.g. due to a system crash, are always detected.

DS, CR	yDelivered service. Check result.
CS, MF	yCorrect service. Malfunction.
DM, CSC	yDetected malfunction. Correct service detected as correct.
NDM, PM	yNot detected malfunction, phantom malfunction.

1.64 Monitoring Parameters



DR	M	Monitoring-MF
CS	CSC	—
CS	PM	Phantom-MF
MF	DM	—
MF	NDM	Masking

Malfunction coverage, proportion of detectable malfunctions:

$$MC = \left.\frac{\#DM}{\#MF}\right|_{\text{SLC}} \tag{1.26}$$

Malfunction permeability as a counter frequency:

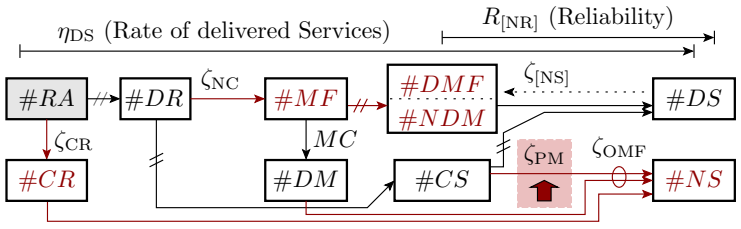
$$MP = \left.\frac{\#NDM}{\#MF}\right|_{\text{SLC}} \tag{1.27}$$

Phantom malfunction rate, the proportion of correct service operations that are classified as malfunctions:

$$\zeta_{\text{PM}} = \left.\frac{\#PM}{\#CS}\right|_{\text{SLC}} \tag{1.28}$$

MC, MT	yMalfunction coverage. Malfunction permeability.
ζ_{PM}	yPhantom malfunction rate.

1.65 CVA Graph without Repeated Request

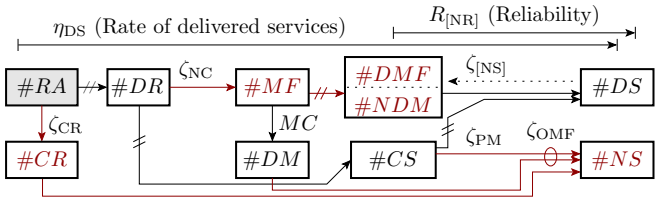


In addition to (slide 1.36), a new edge for phantom malfunctions:

- Malfunctions (MF) are with a frequency of MC detected (DM) and otherwise undetected malfunction (NDM).
- Unless a new request, crashes (CR), detected malfunctions (DM) and phantom malfunctions (MF) are non-delivered services (NS).

ζ_{CR}, ζ yCrash rate. Malfunction rate.
MC, ζ_{PM} yMalfunction coverage. Phantom malfunction rate.
$\zeta_{[NC/NS]}$ yMalfunction rate without checks / with no service for identified problems.
$R_{[NS]}$ yReliability without request (no service) in case of problems.
$\eta_{DS[NS]}$ yDelivery rate without request (no service) in case of problems.

1.66 Delivery Rate without Re-Requests



$$\eta_{DS} = \left.\frac{\#DS}{\#RA}\right|_{ACR} = (1-\zeta_{CR})\cdot(\zeta\cdot(1-MC)+(1-\zeta)\cdot(1-\zeta_{PM}))$$
$$= (1-\zeta_{CR})\cdot\left(\cancel{\zeta}-\zeta\cdot MC+1-\cancel{\zeta}-\zeta_{PM}+\cancel{(\zeta\cdot\zeta_{PM})}^{*}\right) \quad (1.29)$$

With the rate of observable malfunctions,

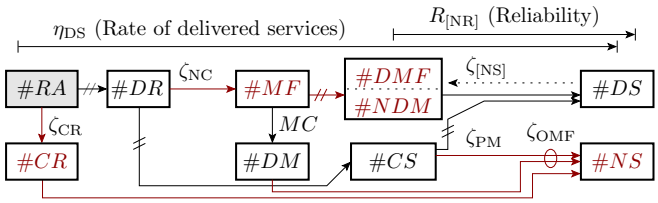
$$\zeta_{OMF} = \zeta_{PM} + \zeta\cdot MCrate \quad (1.30)$$

which is greater than that of (eq. 1.11) by ζ_{PM}, the following still holds:

$$\eta_{DS} = (1-\zeta_{CR})\cdot(1-\zeta_{OMF}) \quad (1.12)$$

* Quadrat sehr kleiner Werte.

1.67 Reliability without Re-Requests



With (eq. 1.30), the previous equations for the rate of delivered malfunctions and reliability also apply,

$$\zeta_{[NS]} = \frac{\zeta_{[NC]}\cdot(1-MC)}{1-\zeta_{OMF}} \quad (1.13)$$

$$R_{[NS]} = \frac{R_{[NC]}\cdot(1-\zeta_{OMF})}{(1-MC)} \quad (1.14)$$

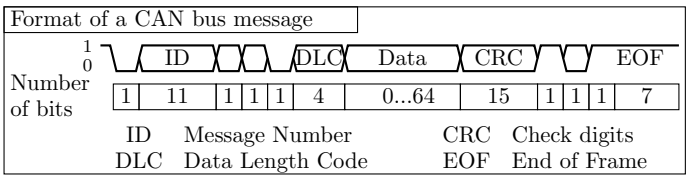
and for a low rate of observable malfunctions $\zeta_{OMF} \ll 1$:

$$R_{[NS]} = \frac{R_{[NC]}}{1-MC} = \frac{R_{[NC]}}{MP} \quad (1.15)$$

$\zeta_{[NC/NS]}$ yMalfunction rate without checks / with no service for identified problems.
$R_{[NC/NS]}$ yReliability without monitoring / when skipping in case of detected problems.
ζ_{OMF} yRate of observable malfunctions.
MC, MT yMalfunction coverage. Malfunction permeability.

2.2 Format Checks

1.68 Format and Value Checks



A service includes data embedded in a format:

- Format: value-independent attributes (value ranges, ...).
- Data: Attributes related to value representation.

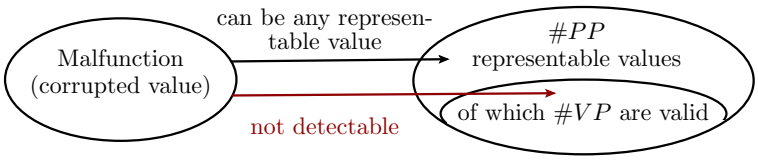
Classification of digital service monitoring:

1. Format Checks: Check only value-independent characteristic. Service results with format errors are always incorrect. Results with the correct format may still contain incorrect values. Check for admissibility only.
2. Value Checks: (Additional) verification of values.

Format Checks are easier to implement and often result in higher MC and lower phantom malfunction rates than value checks.

1.69 Information Redundancy

Format checks (error-detecting codes, check digits, valid range checks, etc.) are based on information redundancy.



Malfunction coverage tends to be higher the smaller the proportion of valid bit patterns is. If all possible pattern falsifications are equally likely, all invalid patterns are identified as invalid and all valid patterns are classified as valid:

$$\mu_{\mathrm{MC}} = 1 - \frac{\#VP}{\#PP} \tag{1.31}$$

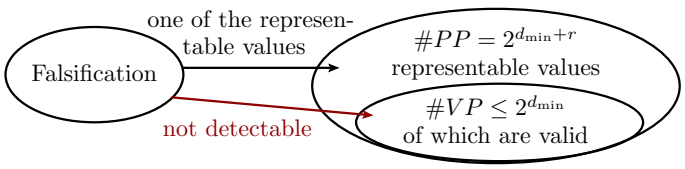
$$\mu_{\mathrm{MP}} = \frac{\#VP}{\#PP} \tag{1.32}$$

$\#VP$ Number of valid bit patterns.
$\#PP$ Number of presentable bit patterns.
μ_{MC} yExpected malfunction coverage of monitoring.

1.70 Redundant Bits

Suppose that d_{min} data bits are sufficient to distinguish all possible values. When represented using r additional (redundant) bits:

$$b = r + d_{\mathrm{min}}$$

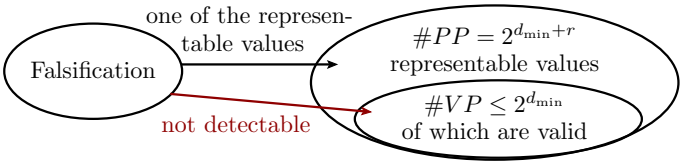


$$\mu_{\mathrm{MC}} = 1 - \frac{\#VP}{\#PP} \geq 1 - \frac{2^{d_{\mathrm{min}}}}{2^{d_{\mathrm{min}}+r}}$$

Minimum coverage for malfunctions:

$$\mu_{\mathrm{MC}} = 1 - 2^{-r} \tag{1.33}$$

$\#VP, \#PP$ yNumber of valid bit patterns. Number of presentable bit patterns.
μ_{MC}, r yExpected malfunction coverage. Number of redundant bits.
d_{min}, b yRequired number of bits to differ between all permissible values. Total number of bits.
μ_{MC}, r yExpected malfunction coverage. Number of redundant bits.



Maximum expected malfunction permeability:

$$\mu_{\text{MT}} = 2^{-r} \tag{1.34}$$

r	10	20	30
μ_{MP}	10^{-3}	10^{-6}	10^{-9}

If all valid values are recognized as valid, there will be no phantom malfunctions during format checks:

$$\zeta_{\text{Phan}} = 0 \tag{1.35}$$

$\#VP, \#PP$ yNumber of valid bit patterns. Number of presentable bit patterns.
$d_{\min}$, b yRequired number of bits to differ between all permissible values. Total number of bits.
r yNumber of redundant bits.
μ_{MP} yExpected malfunction permeability.
ζ_{PM} yPhantom malfunction rate.

1.72 Reliability Improvement

Reliability improvement through **ideal format checking** with **r** redundant bits. No re-request, low rate of observable malfunctions $\zeta_{\text{OMF}} \ll 1$:

(1.33) $\mu_{\text{MC}} = 1 - 2^{-r}$
(1.15) $R_{[\text{NS}]} = \frac{R_{[\text{NC}]}}{1-MC} = \frac{R_{[\text{NC}]}}{MP}$

$$R_{[\text{NS}]} = 2^r \cdot R_{[\text{NC}]} \tag{1.36}$$

Ideal: Distortion not disproportionately often into valid values. This generally applies to error-correcting codes and check digits used in the transmission and storage of data (sec. 5.3).

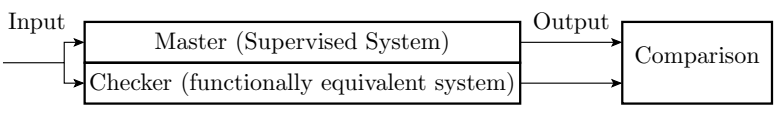
Most **other format checks** (value range, syntax, etc.) do not have the same excellent malfunction coverages, but they still offer a very good cost-benefit ratio (sec. 5.4).

ζ_{OMF} yRate of observable malfunctions.
$R_{[\text{NC/NS}]}$ yReliability without monitoring / when skipping in case of detected problems.
μ_{MC}, μ_{MP} yExpected malfunction coverage. Expected malfunction permeability.
r yNumber of redundant bits.

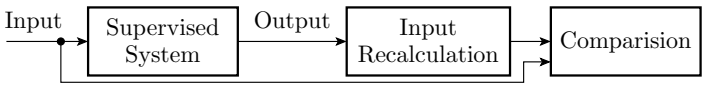
2.3 Value checks, diversity

1.73 Universal Value-Checking Methods

- **Duplication and Comparison** (Master-Checker Principle). Applicable to all deterministic services.

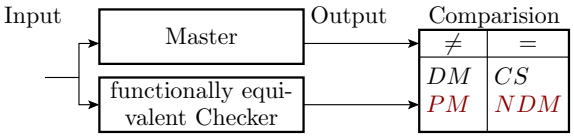


- **Loop Test** (input recalculation and comparison), e.g., monitoring transmission by receiving data and comparing the received data with the transmitted data. Only for reversible, one-to-one functions.



- **Proof of Correctness**: For example, when searching for a path through a graph from A to B , all paths from A to B are valid solutions. Prerequisites for Use: Availability of a correctness condition whose satisfaction proves correctness.

1.74 Properties of Doubling and Comparison



Identical incorrect results from the master and the checker are masked. We want to define the expected malfunction coverage as the diversity rate (see next slide):

$$\mu_{\mathrm{MC}} = \eta_{\mathrm{Div}} \tag{1.37}$$

The phantom malfunction rate is the rate of checker malfunctions without the same master malfunction occurring simultaneously:

$$\zeta_{\mathrm{PM}} = \eta_{\mathrm{Div}} \cdot \zeta_{[\mathrm{Chk}]} \tag{1.38}$$

DM, PM yDetected malfunction. Phantom malfunction.
CS, NDM yCorrect service. Undetected malfunction.
η_{Div} yDiversity rate, defined as fault coverage by doubling and comparison.
$\zeta_{[\mathrm{Mst/Chk}]}$ yMalfunction rate: Master/Checker.

1.75 Diversity and Diversity Rate

Diversity in IT systems describes the variety of problem effects that arise during multiple calculations. In practice, the following applies:

- Problems (malfunctions, crashes) are very rare, and
- There are many ways to falsify data.

Simultaneous identical effects are **practically impossible** due to chance. Coincidence always has a common cause:

- the same incorrect inputs, the same faults,
- failure of shared hardware, ...

The **diversity rate** η_{Div}, defined as the malfunctions coverage by duplication and comparison is also the **correction rate** by repeated calculation. The correction rate for crashes and detected malfunctions—and thus the diversity—can therefore be determined by counting the number of attempted and successful correction attempts.

1.76 Gain in Reliability, observable MF, ...

(1.14) $$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]} \cdot (1-\zeta_{\mathrm{OMF}})}{(1-MC)}$$

(1.12) $$\eta_{\mathrm{DS}} = (1-\zeta_{\mathrm{CR}}) \cdot (1-\zeta_{\mathrm{OMF}})$$

(1.30) $$\zeta_{\mathrm{OMF}} = \zeta_{\mathrm{PM}} + \zeta \cdot MC$$

Using $\mu_{\mathrm{MC}} = \eta_{\mathrm{Div}}$ and $\zeta_{\mathrm{PM}} = \eta_{\mathrm{Div}} \cdot \zeta_{[\mathrm{Chk}]}$, the rate of observed malfunctions is the sum of the rates of diverse master and checker malfunctions:

$$\zeta_{\mathrm{OMF}} = \eta_{\mathrm{Div}} \cdot \left(\zeta_{[\mathrm{Mst}]} + \zeta_{[\mathrm{Chk}]}\right) \tag{1.39}$$

The non-delivery rate is the sum of the rates for crashes plus the diverse master and checker malfunctions:

$$\eta_{\mathrm{DS}} = (1-\zeta_{\mathrm{CR}}) \cdot \left(1-\eta_{\mathrm{Div}} \cdot \left(\zeta_{[\mathrm{Mst}]} + \zeta_{[\mathrm{Chk}]}\right)\right)$$
$$1-\eta_{\mathrm{DS}} \approx \zeta_{\mathrm{CR}} - \eta_{\mathrm{Div}} \cdot \left(\zeta_{[\mathrm{Mst}]} + \zeta_{[\mathrm{Chk}]}\right) \tag{1.40}$$

Reliability gain inversely proportional to the non-diversity rate:

$$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]} \cdot (1-\zeta_{\mathrm{OMF}})}{(1-\eta_{\mathrm{Div}})} \tag{1.41}$$

1.77 Diversity of Identical Systems

Dual calculations and comparisons using the same hardware and software detect virtually all malfunctions caused by disturbances (interference), but hardly any caused by faults.

The diversity rate is equal to the percentage of malfunctions caused by disturbances. It can be determined by repeating with the same system after each issue.

Reduction of malfunction rates to no more than those caused by faults:

$$\zeta \leq \zeta_{\mathrm{F}}$$

Increasing reliability to at least the fault-related partial reliability:

$$R \geq R_{\mathrm{F}}$$

ζ, ζ_{F} yOverall failure rate. Failure rate due to faults.

R, R_{F} yOverall reliability. Partial reliability by faults.

1.78 Expanded Diversity

Structural and organizational measures to increase diversity by addressing common root causes of problems:

Expanded Diversity	Design and organizational measures	Increase in the diversity rate for
Hardware-Diversität	Execution on various hardware platforms	Manufacturing defects, failures
Hardware Design Diversity	Independently designed hardware	In addition, hardware design issues
Syntactic Diversity	Different translation	In addition, software translation issues
Software Design Diversity	Independent Programming	In addition, software design faults
Diverse Uses	Different Sevice Request*	Plus input errors

HW, SW, SR yHardware. Software. Service request.

* Often deviating target values. Not suitable for multiple calculations and comparisons.

1.79 Software Design Diversity

Detecting malfunctions caused by software faults requires that

- the entire software design must be implemented at least twice,
- by separate teams with no communication between them,
- based on a common specification, ...

The initial, euphoric belief that this approach would ensure diversity in the face of all faults, except those specified in the requirements, has not been confirmed. Direct or indirect communication among development teams regarding the interpretation of the requirements—during testing, etc.—brings commonalities into the designs. Human tend to repeat mistakes, ... Achievable diversity rate according to [3] max.

$$\eta_{\text{Div}} \leq 90\%$$

According to (eq. 1.41), the reliability improvement is at most 10. A control system with $r = 10$ bits of information redundancy achieves, according to Eq. 1.33, an improvement of 10^3 with much less design effort and without phantom malfunctions.

1.80 Diverse Use of Services

Real-world IT systems can solve most tasks in different ways (functional redundancy). With other solutions, faults have different effects—or in some cases, no effect at all.

Diverse usage suggests a much higher degree of diversity than repetition with the same inputs. However, the correct results often differ, making it **unsuitable for duplication and comparison**.

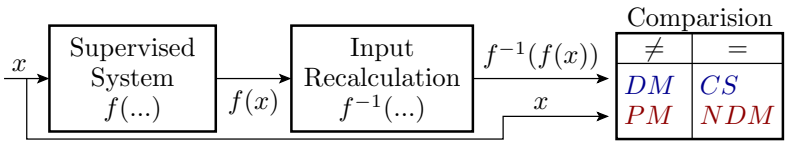
Standard procedure for manual correction by resubmitting the request. Users intuitively try alternative solutions until the problem is resolved (slide 1.105).

Manual testing effort, estimated to be proportional to the non-delivery rate

(1.40) $$1 - \eta_{\text{DS}} \approx \zeta_{\text{CR}} - \eta_{\text{Div}} \cdot \left(\zeta_{[\text{Mst}]} + \zeta_{[\text{Chk}]}\right)$$

making it an important measure of the costs of IT deployment.

1.81 Loop Test Properties



Since $f(\ldots)$ and $f^{-1}(\ldots)$ differ in the algorithm, even without additional constructive and organizational measures, greater diversity in the effects of identical faults can be expected, and thus higher fault coverage and reliability.

The loop test is applicable only if an inverse function $f^{-1}(\ldots)$ can be constructed for $f(\ldots)$, ideally a simpler one, such as squaring to verify the square root calculation.

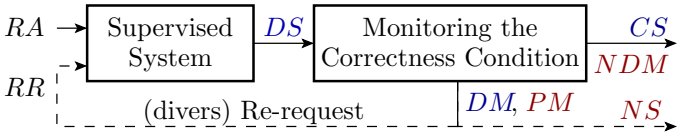
$f(\ldots)$ yFunction of the system to be monitored.

$f^{-1}(\ldots)$ yInverse function.

DM, PM yDetected malfunction. Phantom malfunction.

CS, NDM yCorrect service. Undetected malfunction (product fault).

1.82 Proof of Correctness



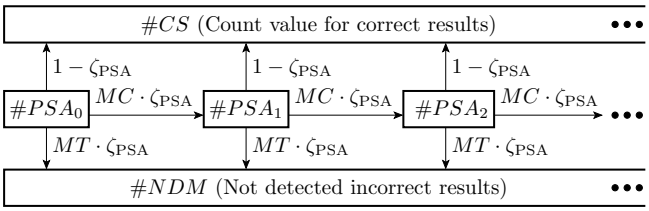
If a sufficient control condition for proving correctness exists (is specified):

- Sorting a list ⇒ The list is sorted and contains all elements,
- Find a path through a graph ⇒ Check that the path is valid,
- Search test for fault detection ⇒ Fault simulation, ...

The malfunction permeability MP of the check is often very low, but a search algorithm that »tries until correctness is proven« also tries to pass false results through the check on average μ_{Trl} times.

RA, DS yAccepted request. Delivered service.
CS, NDM yCorrect service. Undetected malfunction (product fault).
DM, PM yDetected malfunction. Phantom malfunction.
NS, RR yCancellation with no service. Renewed request.

1.83 Random Search with Proof of Correctness



Each count value $\#PSA_i$is with a rate of

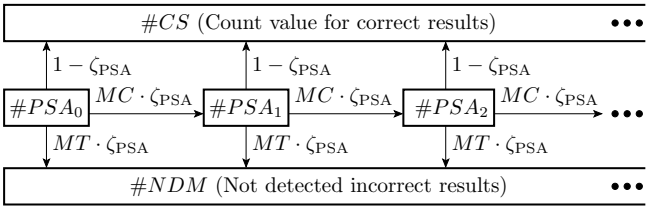
- $1-\zeta_{\mathrm{PSA}}$ a correct result $\#CS$,
- $MC\cdot\zeta_{\mathrm{PSA}}$ another attempt to find a solution $\#PSA_{i+1}$ and
- $(1-MC)\cdot\zeta_{\mathrm{PSA}}$ an undetectable malfunction $\#NDM$.

The fulfillment rate per solution request is one:

$$\eta_{\mathrm{DS}}=\frac{\#CS+\#NDM}{\#PSA_0}=(1-\eta_{\mathrm{PSA}}+(1-MC)\cdot\zeta_{\mathrm{PSA}})\cdot\sum_{i=0}^{\infty}(MC\cdot\zeta_{\mathrm{PSA}})^i=1$$

$$\stackrel{(\mathrm{SGS})}{=}\frac{1-MC\cdot\zeta_{\mathrm{PSA}}}{1-MC\cdot\zeta_{\mathrm{PSA}}}=1$$

SGS ySummation formula of the geometric series: $\sum_{n=0}^{\infty}a_0\cdot q^n=\frac{a_0}{1-q}$.



Average number of trials required for this:

$$\mu_{\mathrm{Trl}}=\sum_{i=0}^{\infty}\eta_{\mathrm{PSA}}^i=\frac{\sum_{i=0}^{\infty}\#DR_i}{\#DR_0}=\sum_{i=0}^{\infty}(MC\cdot\zeta_{\mathrm{PSA}})^i\stackrel{(\mathrm{SGS})}{=}\ldots$$

$$\mu_{\mathrm{Trl}}=\frac{1}{1-MC\cdot\zeta_{\mathrm{PSA}}}\tag{1.42}$$

Malfunction rate of the generated solutions:

$$\zeta=\frac{\#NDM}{\#PSA_0}=(1-MC)\cdot\zeta_{\mathrm{PSA}}\cdot\sum_{i=0}^{\infty}(MC\cdot\zeta_{\mathrm{PSA}})^i\stackrel{(\mathrm{SGS})}{=}\frac{(1-MC)\cdot\zeta_{\mathrm{PSA}}}{1-MC\cdot\zeta_{\mathrm{PSA}}}$$

$$\zeta=\mu_{\mathrm{Trl}}\cdot MT\cdot\zeta_{\mathrm{PSA}}\tag{1.43}$$

ζ_{PSA}, μ_{Trl} yProblem-solving rate. Expected number of trials.
$\#PSA_i$, ζ yCount for solution attempt i. Overall failure rate.
MC, MT yMalfunction coverage. Malfunction permeability.

1.87 Extensive Search, Check Quality, AI

(1.42) $$\mu_{\mathrm{Trl}} = \frac{1}{1-MC\cdot\zeta_{\mathrm{PSA}}}$$

(1.43) $$\zeta = \mu_{\mathrm{Trl}} \cdot MT \cdot \zeta_{\mathrm{PSA}}$$

The rate of malfunctions among all the solutions that have been put through the check increases proportionally with the number of trials required to get the solution.

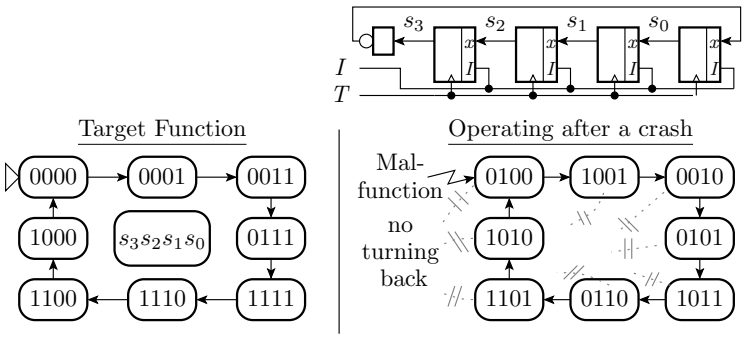
AI (artificial intelligence) services sift through large amounts of data using vaguely defined search criteria. It is less surprising that some of the answers are incorrect than that there are any useful results at all.

Reliable AI requires good strategies for finding solutions with few attempts and, above all, checks that (almost) never allow incorrect solutions to slip through.

ζ_{PSA}, μ_{Trl} yProblem-solving rate. Expected number of trials.
MC, MT yMalfunction coverage. Malfunction permeability.
ζ yOverall failure rate.

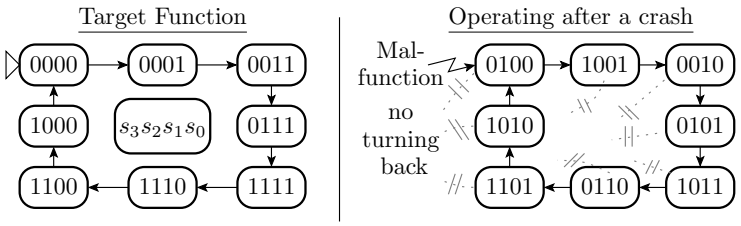
2.4 Time Monitoring

1.89 Crashes and Time Tracking



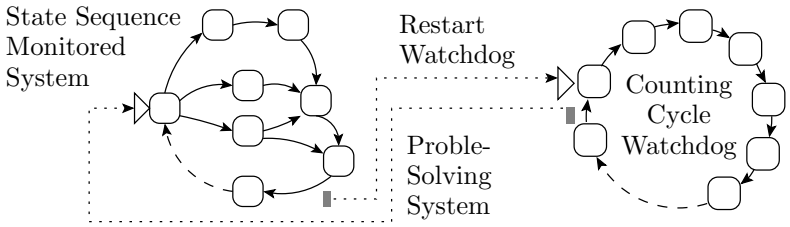
- Automata and programs use only a small fraction of the $2^{\#SB}$ possible states ($\#SB$ – number of state bits).
- The 4-bit Johnson counter cycles through 8 of the 16 states.
- ...

I, T, s_i yInitialisation signal. Clock signal. State bit i.



- The remaining (redundant) 8 states form a cycle from which the system will never exit.
- A transition into invalid states that cannot be exited without reinitialization is a crash.
- Complex hardware and software have millions of state bits and an unmanageable number of ways to crash.
- **A crash** (CR) **is always detected** by that the system no longer delivers results (DR).
- Crash handling: time monitoring, termination, performance waiver, or retry. The success rate of the retry is equal to the diversity rate of crashes (see slide 1.75).

1.91 Watchdog



A watchdog is a timer that must be reset by the monitored system before it overflows. An overflow triggers a restart, which returns the system to a valid state.
A hardware watchdog is a counter register in the processor

- Once started, it cannot be turned off by the system itself
- and a machine command to reset it.

In the event of an overflow, a processor reset or an interrupt with a problem-solving routine.

2.5 Correction

1.92 Dealing with detected problems

With a sufficiently high fulfillment rate, service waiver, or manual attempts to obtain the desired services:

1. Repeated identical requests to the same service provider.
2. A modified request to the same provider, or
3. another provider.

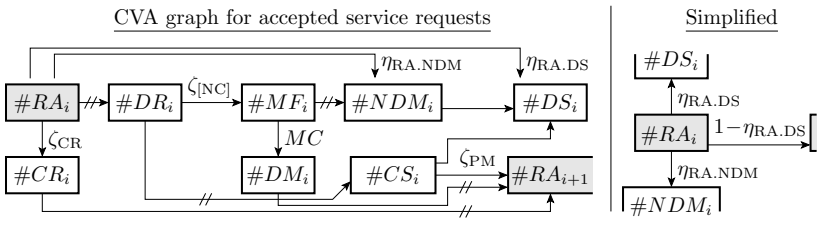
(2) and (3) diverse servic uses, which are difficult to automate.

Automated processing for (1) can be divided into:

- New requests until fulfillment and
- a limited number of new requests (typically, max. one).

Error-correcting codes provide another, entirely different type of automatic correction, though only for data transmission and storage.

1.93 CVA Graph Simplification

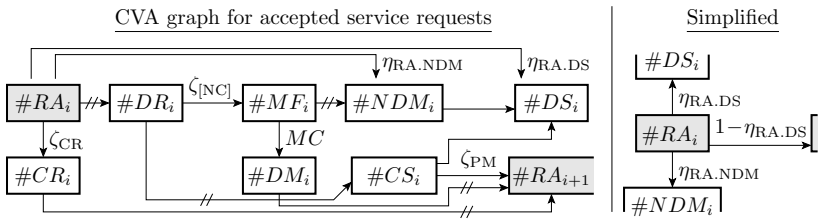


Rate of delivered services per accepted service request

$$\frac{\#DS_i}{\#RA_i} = \eta_{\text{RA.DS}} = (1-\zeta_{\text{CR}}) \cdot \big(\zeta_{[\text{NC}]} \cdot (1-MC) + (1-\zeta) \cdot (1-\zeta_{\text{PM}})\big)$$

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
CR_i, DR_i yOverall failure rate.
DM_i, CS_i yCrash at request i. Result delivered for request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.

ζ_{CR}, $\zeta_{[\text{NC}]}$ yCrash rate. Malfunction rate without monitoring.
MC, ζ_{PM} yMalfunction coverage. Phantom malfunction rate.



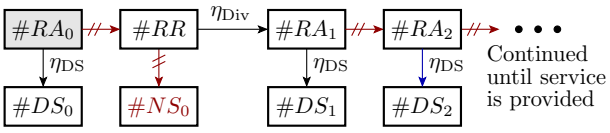
Rate of undetected (delivered) malfunctions per accepted service request (See also slide 1.66):

$$\frac{\#NDM_i}{\#RA_i} = \eta_{\text{RA.DS}} = (1-\zeta_{\text{CR}}) \cdot \left(\zeta_{[\text{NC}]} \cdot (1-MC)\right)$$

For the analysis with repeated requests, replace the left CVA graph for each single service request **with the simplified graph on the right.**

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
CR_i, DR_i yOverall failure rate.
DM_i, CS_i yCrash at request i. Result delivered for request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.

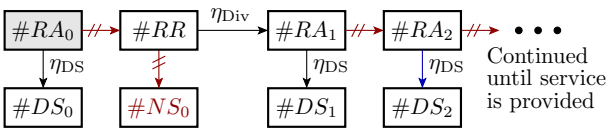
1.95 Repeat Until Delivery [RR]



The first accepted request (RA_0) is a service provided (DS_0) at the rate $\eta_{\text{RA.DS}}$; otherwise, it is a repeat request until a result is provided (RR).

If the repeat result is identical (non-diverse), terminate NS_0 because no correction is possible. Otherwise, accepted diverse request (RA_1), which, with $\eta_{\text{RA.DS}}$, is a provided service (DS_1).

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.

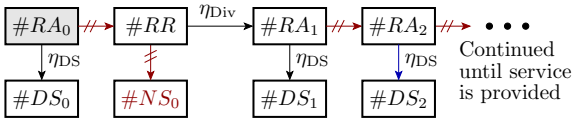


If not delivered, further (diverse) retry attempts are made until delivery. The accepted request (RA_0) and all diverse retry results (RA_i with $i \geq 1$) are not detected (delivered) malfunctions (NDM_i) with the rate $\eta_{\text{RA.NDM}}$.

For retry attempts, it is assumed that the request is always accepted—either hardware and service availability are set to one, or that attempts without an accepted request do not count.

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.

1.97 Total Delivery Rate



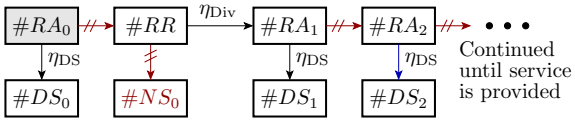
The delivery rate is one minus the rate of NS_0 counts:

$$\eta_{\text{DS[RR]}} = 1 - \left.\frac{\#NS_0}{\#RA_0}\right|_{\text{SLC}}$$
$$\eta_{\text{DS[RR]}} = 1 - (1 - \eta_{\text{RA.DS}}) \cdot (1 - \eta_{\text{Div}}) \tag{1.44}$$

Delivery for all detectable divers issues (crashes, malfunctions). The other issues always occur in the same way on request and therefore cannot be corrected.

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.
$\eta_{\text{DS[RR]}}$ yDelivery rate, in the case of repeated requests, until delivery.

1.98 Expected Number of Trials



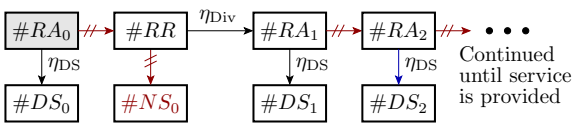
Sum of all $\#RA_i$ counts per service request $\#RA_0$:

$$\mu_{\text{Trl}} = \left.\frac{\sum_{i=0}^{\infty} \#RA_i}{\#RA_0}\right|_{\text{ACR}} = 1 + \eta_{\text{Div}} \cdot \sum_{i=1}^{\infty} (1 - \eta_{\text{RA.DS}})^i \overset{\text{(SGS)}}{=} 1 - \eta_{\text{Div}} + \frac{\eta_{\text{Div}}}{\eta_{\text{RA.DS}}}$$

For perfect diversity ($\eta_{\text{Div}} = 1$, e.g., problems caused solely by disturbances), the expected number of trials is equal to the reciprocal of the delivery rate per trial (see Eq. 1.42). A low delivery rate results in a large number of trials.

RA_i, DS_i yRequest i accepted. Service provided for request ie.
NDM_i yNot detected (delivered) malfunction for accepted request i.
$\eta_{\text{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\text{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.
μ_{Trl} yExpected number of trials.

1.99 Reliability



Compared to a scenario without retries, the rate of undetected malfunctions increases by the expected number of trials:

$$\eta_{\mathrm{NDM[RR]}} = \left.\frac{\sum_{i=0}^{\infty}\#NDM_i}{\#RA_0}\right|_{\mathrm{SLC}} = \eta_{\mathrm{RA.NDM}} \cdot \left(\underbrace{1-\eta_{\mathrm{Div}}+\eta_{\mathrm{Div}}\cdot\sum_{i=1}^{\infty}(1-\eta_{\mathrm{RA.DS}})}_{\mu_{\mathrm{Trl}}} \right)$$

Reliability as the delivery ratio to the rate of undetected malfunctions...

$\eta_{\mathrm{MDM[RR]}}$ yOverall rate of undetected malfunctions.

1.100 Reliability Loss due to Repetition

Reliability is the ratio of the delivery rate to the rate of undetected malfunctions:

$$R_{\mathrm{[RR]}} = \frac{\eta_{\mathrm{DS[RR]}}}{\eta_{\mathrm{NDM[RR]}}} = \frac{1-(1-\eta_{\mathrm{RA.DS}})\cdot(1-\eta_{\mathrm{Div}})}{\eta_{\mathrm{RA.NDM}}\cdot\mu_{\mathrm{Trl}}}$$

When the delivery rate $\eta_{\mathrm{NDM[RR]}}$ is close to one, the reliability with repetition is approximately the quotient of the reliability without repetition and the expected number of trials until delivery:

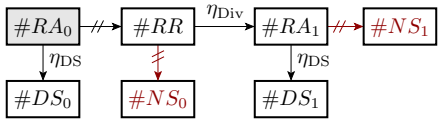
$$R_{\mathrm{[RR]}} = \frac{R_{\mathrm{[NS]}}}{\mu_{\mathrm{Trl}}} \tag{1.45}$$

With a high delivery rate and few problems, regardless of whether repeat attempts are made:

$$R = \frac{R_{\mathrm{[NC]}}}{MP} \tag{1.46}$$

$R_{\mathrm{[RR]}}$ yReliability, in the case of repeated requests, until delivery.
$\eta_{\mathrm{DS[RR]}}$ yDelivery rate, in the case of repeated requests, until delivery.
$\eta_{\mathrm{MDM[RR]}}$ yOverall rate of undetected malfunctions.
$\eta_{\mathrm{RA.DS}}$ yThe rate at which accepted requirements are delivered as services.
$\eta_{\mathrm{RA.NDM}}$ yThe rate of accepted requirements that are undetected faults.
μ_{Trl} yExpected number of trials.
$R_{\mathrm{[NS]}}$ yReliability without request (no service) in case of problems.

1.101 At most one Retry [SR]



Limiting the maximum number of repetitions reduces the expected number of repetitions. This improves reliability but reduces the delivery rate. Delivery rate with one retry:

$$\eta_{\mathrm{DS[SR]}} = \left.\frac{\#DS_0+\#DS_1}{\#RA_0}\right|_{\mathrm{SLC}} = \eta_{\mathrm{RA.DS}} + (1-\eta_{\mathrm{RA.DS}})\cdot\eta_{\mathrm{Div}}\cdot\eta_{\mathrm{RA.DS}}$$

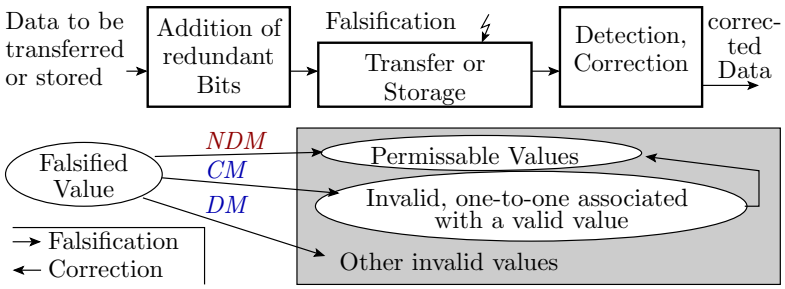
$$\eta_{\mathrm{DS[SR]}} = 1 - \underbrace{(1-\eta_{\mathrm{RA.DS}})\cdot(1-\eta_{\mathrm{Div}})}_{\text{nicht erbrachte nicht diversitäre Leitungen}} - \underbrace{\eta_{\mathrm{Div}}\cdot(1-\eta_{\mathrm{RA.DS}})^2}_{0 \text{ for } \eta_{\mathrm{RA.DS}}\to 1} \tag{1.47}$$

With a high delivery rate per request, as for »request-to-delivery« practically for all diverse problems service will be delived:

(1.44) $$\eta_{\mathrm{DS[RR]}} = 1-(1-\eta_{\mathrm{RA.DS}})\cdot(1-\eta_{\mathrm{Div}})$$

$\eta_{\mathrm{DS[SR]}}$ yDelivery rate in case of a maximum one repeated request.

1.102 Error-correcting Codes (data loss)



Correcting corrupted data after transfer and storage:

- Adding additional (redundant) bits before transmission or storage, more than for error-detecting codes.
- Replace incorrect, correctable values with the corresponding correct values.

For details on implementation, see the section 5.3.5.

2.6 Automated/Manual

1.103 Acting on detected problems

Automated response to system crashes and detected malfunctions:

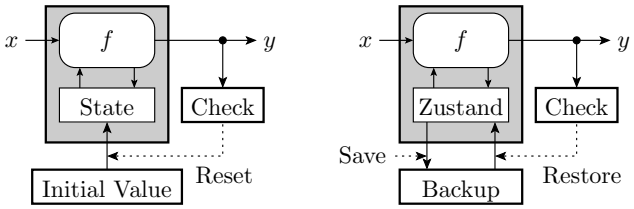
- Termination due to time-out (slide 1.89).
- Establishing a safe system state (slide 1.49).
- Error message to users for manual handling (slide 1.105) and to the manufacturer for rectifying the fault (sec.2.3.6).
- Restoring functionality: If a hardware fault is suspected, repair/reconfigure. Allways reinitialise.
- New request or cancellation without provision of services.

Manual action in the event of no automatic service provision::

- Waiver. Identical or different new request to the same system or an alternative system.
- Search for the cause, rectification and resubmission, ...

A significant factor in the functionality of software and in the amount of manual work involved in using IT services.

1.104 Re-initialisation



In the event of a malfunction, internal data is often corrupted. To restore the system to a functional state, the internal data must be reinitialised with valid values:

- Static reinitialisation (reset): fixed initial state,
- Dynamic reinitialisation: Regular backups during operation. Loading the most recent backup upon detection of an MF.

Often, only data that cannot be easily recalculated is backed up – in the case of editors, logistics systems, databases, etc. – specifically, the data entered since the last full backup.

x, f, y yInput. Function of the system. Output.

1.105 Manual workaround

If the system fails to respond following a crash or a detected malfunction – whether or not automatic retry attempts are made – a user will generally try to solve the task manually in a different way. The error messages provide guidance.

User new request, ideally until success:

- with different input data,
- different ways and orders of entry,
- Options for the calculation method, the output, ...

This exploits the functional redundancy of IT systems. There are usually many ways to solve a problem, and the impact of faults varies depending on the approach taken; in the best-case scenario, there are no effects at all (slide 1.80).

Over time, users learn to avoid or get round the observed problems (see later: »Maturation of user behaviour«, slide 2.102).

1.106 Problem Prevention

particularly those that are difficult to deal with or pose a safety risk:

- Resistance to interference, fault isolation,
- Avoiding the use of unsafe descriptions,
- Guidelines for use and workplace safety,
- Security measures, disconnection from the internet,
- The quiescent current principle, ...

Quiescent current principle: system desig so that, in the event of anticipated problem, a safe state is automatically established:

- Past Railway signalling: Blocking of the section of track in the event of a break in the cable used to operate th
- Fire alarm system: triggers an alarm in the event of a broken wire.
- Vehicle brakes: Braking when a brake hose bursts, ...

Summary

1.107 Problems during Operation (𝕃)

Iteration of monitoring

- Time-based monitoring for crashes (watchdog principle),
- Monitoring of formats (values-independent characteristics),
- additional value check.

and response to identified problems:

- Establishing a safe (hazard-free) state,
- Documenting the problem,
- Repair, if necessary.,
- Reinitialisation (dynamic, static),
- Service refusal (NS) or repeat request (RR).

Avoiding problems that are difficult to deal with:

- quiescent current principle,
- disconnection from the internet,

Preventing data loss by means of error-correcting codes.

1.108 Key parameters for monitoring (𝕃)

(1.26) $$MC = \left.\frac{\#DM}{\#MF}\right|_{\text{SLC}}$$ (1.28)

$$\zeta_{\text{PM}} = \left.\frac{\#PM}{\#CS}\right|_{\text{SLC}}$$

Ideal format checking with r redundant bits:

(1.33) $$\mu_{\text{MC}} = 1 - 2^{-r}$$

(1.35) $$\zeta_{\text{Phan}} = 0$$

Other format checks (value range checks, syntax checks, etc.) are difficult to model and have poorer performance, yet often offer a better cost-benefit ratio than value checks.

Value checking by **doubling and comparing**:

(1.37) $$\mu_{\text{MC}} = \eta_{\text{Div}}$$

(1.38) $$\zeta_{\text{PM}} = \eta_{\text{Div}} \cdot \zeta_{[\text{Chk}]}$$

Other value checks (loop test, specified proof of correctness) are only suitable for specific systems, but offer better detection properties.

1.109 Delivery rate (𝔹)

Delivery rate without new requests:

(1.12) $$\eta_{\text{DS}} = (1 - \zeta_{\text{CR}}) \cdot (1 - \zeta_{\text{OMF}})$$

(1.30) $$\zeta_{\text{OMF}} = \zeta_{\text{PM}} + \zeta \cdot MC$$

Repeat until no visible problem remains; this corrects all diverse issues:

(1.44) $$\eta_{\text{DS[RR]}} = 1 - (1 - \eta_{\text{RA.DS}}) \cdot (1 - \eta_{\text{Div}})$$

With a high delivery rate – even when limited to a maximum of one new request – all diverse issues are resolved.

(𝕃) Where problem rates are low, it is sufficient to filter out identified problems or, at most, submit one new request. Dealing with malfunctions by submitting several new requests is only worthwhile where services are provided in a diverse manner and the fulfilment rates per request is very low.

1.111 Improvements in reliability (𝕃)

Without re-request, if problems were identified

(1.14) $$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]} \cdot (1-\zeta_{\mathrm{OMF}})}{(1-MC)}$$

(1.30) $$\zeta_{\mathrm{OMF}} = \zeta_{\mathrm{PM}} + \zeta \cdot MC$$

and a low rate of observable malfunctions $\zeta_{\mathrm{OMF}} \ll 1$ reliability is the reliability without monitoring divided by the malfunction permeability of monitoring:

(1.15) $$R_{[\mathrm{NS}]} = \frac{R_{[\mathrm{NC}]}}{1-MC} = \frac{R_{[\mathrm{NC}]}}{MP}$$

Repeated requests for diverse problems reduce reliability by the expected number of requests:

(1.45) $$R_{[\mathrm{RR}]} = \frac{R_{[\mathrm{NS}]}}{\mu_{\mathrm{Trl}}}$$

With a high delivery rate and few problems, regardless of whether repeat attempts are made:

(1.46) $$R = \frac{R_{[\mathrm{NC}]}}{MP}$$

1.112 Diversity (𝕃)

Diversity: The variety in the effects of problems when the same tasks are solved in different ways.

Diversity rate: Malfunction coverage through duplication and comparison. Correction rate of recalculation. Percentage of malfunctions without a common cause.

Double-checking and comparing using the same hardware and software detects virtually all malfunctions caused by interference, but hardly any caused by faults. Option to check whether the issue is due to an fault or a disturbance.

Extending diversity to

- manufacturing defects and failures: different hardware,
- hardware design faults: independently designed hardware, ...
- Separate software designs based on a common specification achieve up to $\eta_{\mathrm{Div}} \leq 90\%$.

Manual workaround based on diverse use of services.

1.113 Bibliography

References

[1] J. Hartmann. *Analyse und Verbesserung der probabilistischen Testbarkeit kombinatorischer Schaltungen*. PhD thesis, Diss. Unversität des Saarlandes, 1992.

[2] J.C. Laprie. Dependable computing and fault tolerance: Concepts and terminology. In *15th IEEE Int. Symp. on Fault-Tolerant Computing*, 1985.

[3] U. Voges. Software-Diversität und ihre Modellierung - Software-Fehlertoleranz und ihre Bewertung durch Fehler- und Kostenmodelle. In *Informatik-Fachberichte*. Springer, 1989.