

SOFTWARE FAALKANSANALYSE



Refis

system reliability engineering

Refis™ houdt zich bezig met onderzoek, ontwikkeling, advies en opleidingen op het gebied van betrouwbaarheid van informatiesystemen.

Tweede herziene uitgave, oktober 2008.

Alle rechten voorbehouden. Niets uit deze uitgave mag worden verveelvoudigd, opgeslagen in een geautomatiseerd gegevensbestand, of openbaar gemaakt, in enige vorm of op enige wijze, hetzij elektronisch, mechanisch, door fotokopieën, opnamen, of enig andere manier, zonder voorafgaande schriftelijke toestemming van de uitgever.

**Refis V.O.F. System
Reliability Engineering**
Merellaan 5
3722 AK Bilthoven
Tel: +31(0)30 225 36 37
Fax +31(0)30 225 36 49
email info@refis.nl
website www.refis.nl
K.v.K. Utrecht 30185566

SOFTWARE FAALKANSANALYSE

INHOUD

1 SOFTWAREBETROUWBAARHEID	5
1.1 DEFINITIE.....	5
1.2 CONCEPT VAN BETROUWBAARHEIDSGROEI	5
1.2.1 <i>Betrouwbaarheid voor een bepaalde periode in tijd</i>	6
1.2.2 <i>Toename van betrouwbaarheid</i>	6
1.2.3 <i>Betrouwbaarheidsgroei-curve</i>	7
2 BETROUWBAARHEIDSGROEI MODELEN	11
2.1 JELINSKI-MORANDA MODEL	11
2.2 GEOMETRIC MODEL	12
2.3 MUSA BASIC MODEL	12
3 FAALKANSANALYSE	15
3.1 ANALYSE VAN BESCHIKBARE GEGEVENS	15
3.2 CONVERSIE	15
3.2.1 <i>Refis/Laquso SRE-tool</i>	15
3.2.2 <i>Aansluiting op gangbare incidentmanagement- en testtools</i>	16
3.3 MODELSELECTIE.....	16
3.4 BRUIKBAARHEID VAN GEGEVENS	16
3.4.1 <i>Voortschrijdend gemiddelde</i>	17
3.5 UITGANGSPUNTEN VOOR BEREKENING.....	18
3.5.1 <i>Parameter schatting</i>	18
3.5.2 <i>Gegevensbereik</i>	19
3.5.3 <i>Filters</i>	19
3.5.4 <i>Voorspelling</i>	19
3.6 RESULTATEN.....	19
3.6.1 <i>Ingangsdata</i>	20
3.6.2 <i>Voortschrijdend gemiddelde</i>	21
3.6.3 <i>Betrouwbaarheid / Faalkans</i>	22
3.6.4 <i>Cumulatief aantal defects</i>	23
3.6.5 <i>Foutintensiteit</i>	24
3.6.6 <i>Statistische significantie</i>	24
3.7 INTERPRETEREN EN RAPPORTEREN VAN RESULTATEN	25
4 SAMENVATTING	26
4.1 SOFTWAREBETROUWBAARHEID	27
4.2 FAALKANSANALYSE.....	27
BIJLAGE A TIME BETWEEN FAILURE VERSUS FAILURE COUNT PER INTERVAL ...	29
INDEX	31
FIGUREN	31
LITERATUUR	31

1 SOFTWAREBETROUWBAARHEID

1.1 Definitie

Software is, kort gezegd, het geheel van programma-instructies waarmee door een computer bepaalde functies kunnen worden uitgevoerd. Deze functies kunnen het ontvangen en opslaan van gegevens of signalen inhouden, het bewerken daarvan en het zenden of presenteren van al dan niet bewerkte gegevens of signalen. Zolang de software beschikbaar is en zij deze functies conform specificaties uitvoert, is zij betrouwbaar. In praktijk betekent dit dus dat betrouwbare software op elk willekeurig moment en in bekende omstandigheden, bekend of voorspelbaar gedrag vertoont.

Aangezien algemeen is geaccepteerd dat software niet betrouwbaar is, althans niet volledig, is jaren geleden al de behoefte ontstaan om die mate van (on)betrouwbaarheid te kunnen kwantificeren. Als software niet op elk willekeurig moment en niet in alle omstandigheden, het bekende of voorspelbare gedrag vertoont, hoe groot is dan de kans daarop.

Vervolgens zijn in de loop van de jaren daarop verschillende definities van betrouwbaarheid gegeven. Watts Humphrey spreekt in het Capability Maturity Model (CMM) [HUM], over “[...] the ability to perform the intended functions whenever needed”. De International Standardization Organization (ISO) hanteert een container begrip voor betrouwbaarheid [ZEI]: “A set of attributes that bear on the ability of software to maintain its level of performance under stated conditions for a stated period of time”. En Michael R. Lyu introduceert het begrip “time to failure”: “A measure of continuous delivery of the correct service – or, equivalent, of the time to failure” [LYU].

De definitie van betrouwbaarheid van John Musa sluit echter het beste aan op het doel in het kader van faalkansanalyse, het kunnen kwantificeren van betrouwbaarheid:

De waarschijnlijkheid dat een systeem een bepaalde functie foutloos uitvoert onder bepaalde omstandigheden voor een bepaalde periode in tijd.

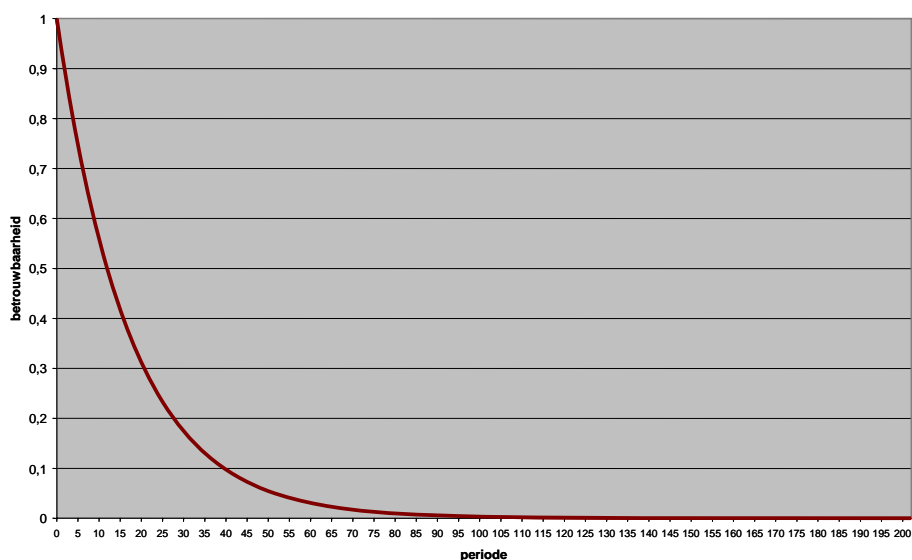
Dit is de “standard definition of reliability” [MUS1]. Toepassing ervan vereist een nauwkeurige omschrijving van het onderdeel, de fouten, de omstandigheden en de periode in tijd.

1.2 Concept van betrouwbaarheidsgroei

De definitie van betrouwbaarheid van Musa, levert direct ook het inzicht in de meeteenheid: waarschijnlijkheid of kans wordt uitgedrukt in een getal tussen 0 en 1 (desgewenst als percentage). In paragraaf 1.2.1 wordt de relatie getoond tussen deze kans en de periode in tijd waarover een uitspraak wordt gedaan. In paragraaf 1.2.2 wordt vervolgens getoond hoe de betrouwbaarheid toeneemt naarmate er meer defects worden gevonden en opgelost. Dit resulteert in de in paragraaf 1.2.3 getoonde betrouwbaarheidsgroei-curven.

1.2.1 Betrouwbaarheid voor een bepaalde periode in tijd

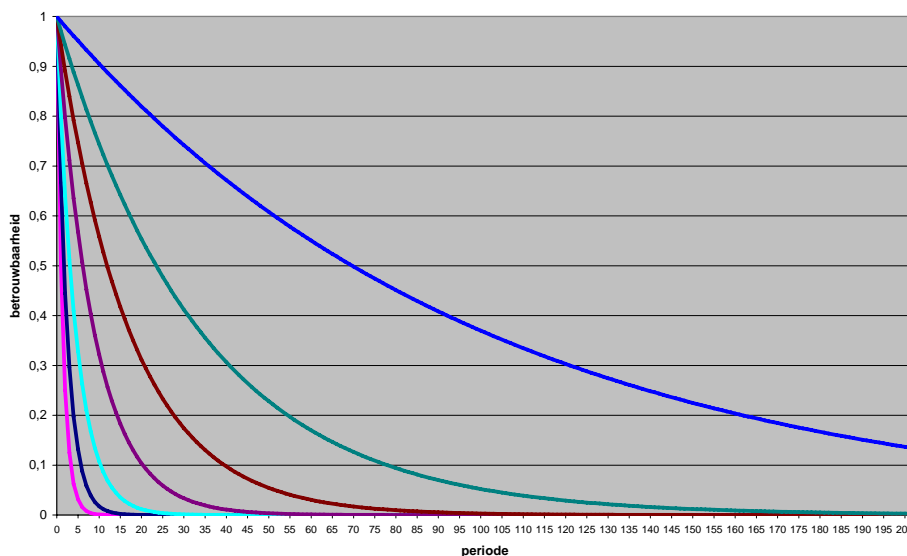
Als de software in een bepaalde tijd zeker niet zal falen is de betrouwbaarheid 1 (of 100%). Als de software in een bepaalde tijd zeker wel zal falen is de betrouwbaarheid 0 (of 0%). Aangezien betrouwbaarheid dus gekoppeld is aan een bepaalde tijd (waarop het systeem operationeel is), is er niet één getal waarin betrouwbaarheid kan worden uitgedrukt. Het zijn er velen, elk voor een bepaalde periode waarover de betrouwbaarheid is gemeten of wordt voorspeld. Zo is de betrouwbaarheid van een willekeurig systeem nagenoeg 1 voor de komende twee seconden en nagenoeg 0 voor de komende tienduizend jaar. De betrouwbaarheidscurve tussen deze twee punten loopt bijvoorbeeld als getoond in *Figuur 1*. Op de x-as is de periode van tijd weergegeven, op de y-as de betrouwbaarheid. De x-as is verdeeld in periodes van 5 tijdseenheden. Welke eenheden dit zijn (seconden, dagen of jaren) maakt voor het principe niet uit.



Figuur 1; betrouwbaarheidscurve

1.2.2 Toename van betrouwbaarheid

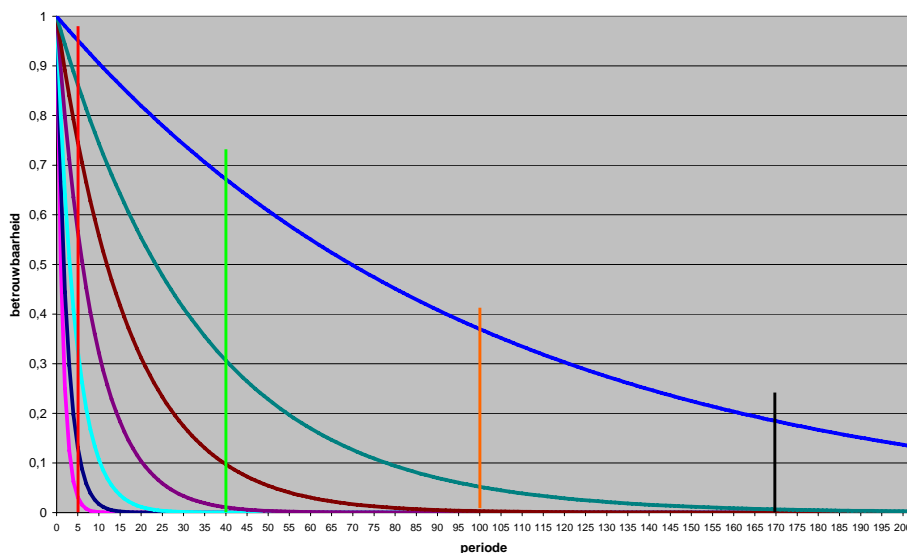
Het uitgangspunt voor de betrouwbaarheidsmodellen is, zoals gezegd, dat de betrouwbaarheid toeneemt met elke fout die gevonden en hersteld wordt. In *Figuur 2* zijn de opeenvolgende betrouwbaarheidscurven van één systeem getoond na het oplossen van 7 individuele defects. Elke curve geeft de betrouwbaarheid weer ná het oplossen van het defect, voor een willekeurig aantal periodes. Deze periodes zijn weergegeven op de x-as, de betrouwbaarheid zelf op de y-as.



Figuur 2; betrouwbaarheidscurves na oplossen van 7 individuele defects

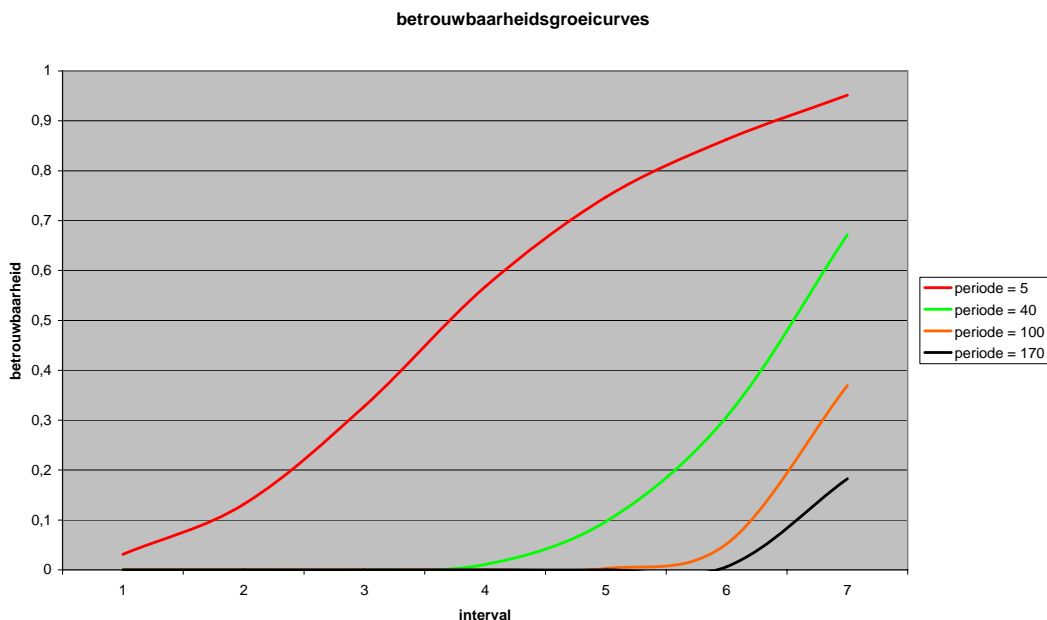
1.2.3 *Betrouwbaarheids-groei-curve*

Men zou nu de betrouwbaarheids-groei kunnen meten voor 4 verschillende periodes in tijd (in *Figuur 3* voor 5, 40, 100 en 170 tijdseenheden). De betrouwbaarheids-groei is het verschil tussen de betrouwbaarheid voor een bepaalde periode na n defects en de betrouwbaarheid voor diezelfde periode na n+1 defects.



Figuur 3; betrouwbaarheids-groei voor 4 periodes

De betrouwbaarheids-groei-curve voor deze vier periodes zien er dan als volgt uit (*Figuur 4*):



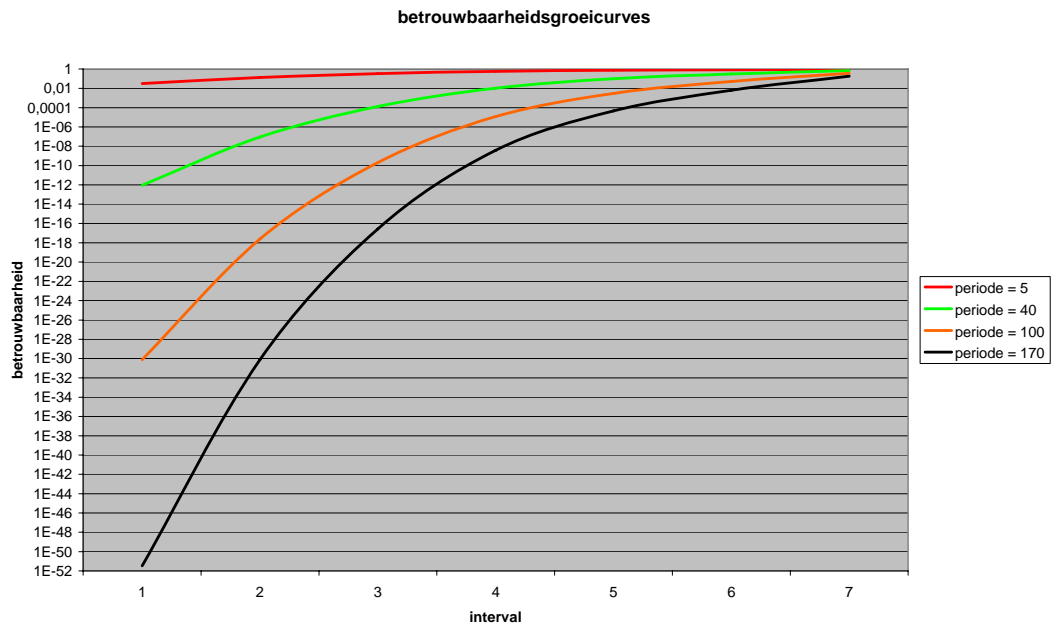
Figuur 4; betrouwbaarheids-groei-curve

De x-as toont de intervallen in het testtraject waarop de fouten zijn gevonden en opgelost (in dit voorbeeld dus 1 defect per interval). De betrouwbaarheid is weergegeven op de y-as. De curve tonen de toename van de betrouwbaarheid na elk interval. Een en ander betekent dat na het vinden en herstellen van 7 fouten, het systeem een betrouwbaarheid voor 5 tijdseenheden heeft van 0,9515 (het eindpunt van de rode lijn), een betrouwbaarheid voor 40 tijdseenheden van 0,6717 (groene lijn), voor 100 eenheden van 0,7419 (oranje) en voor 170 eenheden van 0,3697 (zwart). Desgewenst kan de betrouwbaarheid voor elke willekeurige periode in tijd worden berekend.

De *Figuur 5* toont dezelfde gegevens maar nu op een logaritmische schaal van betrouwbaarheid. Daarin is duidelijk de betrouwbaarheidstoename tot 1 te herkennen voor elke willekeurige periode. Het principe is immers dat de betrouwbaarheid groeit, doch nooit 1 zal worden.

Wanneer het betreffende systeem, de soorten defecten en de betrouwbaarheids-groei-curve voldoen aan één of meerdere beschreven betrouwbaarheidsmodellen, kan niet alleen de betrouwbaarheid van nu worden berekend, maar tevens een voorspelling worden gedaan over de ontwikkeling daarvan voor de komende n defecten of periodes in tijd.

Overigens zal duidelijk zijn dat het in dit voorbeeld gebruikte aantal van 7 defecten statistisch gezien niet erg hoog is om met enige zekerheid vast te stellen of de curve al of niet past binnen een bepaald model.



Figuur 5; betrouwbaarheidsgroecurves logaritmisch

2 BETROUWBAARHEIDSGROEIMODELLEN

Betrouwbaarheidsgroei modellen beschrijven de toename van betrouwbaarheid gebaseerd op het aantal defects dat is opgelost. Daarbij wordt onderscheid gemaakt naar een aantal bepalende kenmerken zoals het soort systeem, het gebruik ervan, de soorten opgetreden defects en de kwaliteit van de correcties. Ter illustratie is een aantal van de bekendste betrouwbaarheidsmodellen hieronder beschreven:

- Jelinski-Moranda model
- Geometric model
- Musa Basic model

In het Refis/Laquso SRE-tool zijn ook Bayesiaanse modellen opgenomen (zoals het model van Littlewood en Verall) [BRA].

2.1 Jelinski-Moranda model

Vroeg in de jaren zeventig werd bij McDonnell Douglas door P.L. Moranda en Z. Jelinski een model ontwikkeld voor het meten van softwarebetrouwbaarheid. Het door hen voorgestelde model wordt nog steeds gebruikt, en is één van de oudste modellen op het gebied van software reliability engineering. Het model van Jelinski-Moranda is onder meer gebaseerd op de volgende aannames:

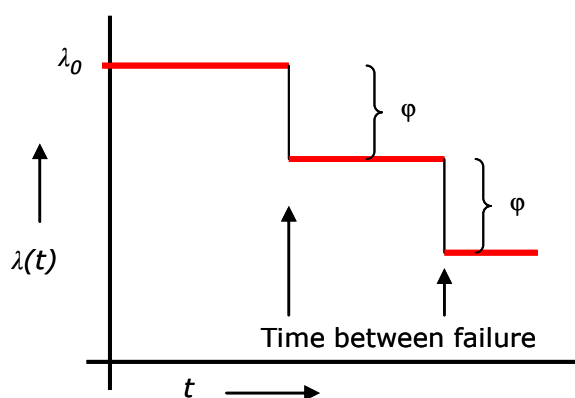
1. De mate waarin fouten optreden is evenredig met het aantal nog aanwezige fouten in de software;
2. De mate waarin fouten optreden is constant in het tijdsinterval tussen twee opeenvolgende fouten;
3. Iedere fout die optreedt wordt onmiddellijk verholpen, en hierbij worden geen nieuwe fouten geïntroduceerd;

De "mate waarin fouten optreden" waar in uitgangspunten 1 en 2 sprake van is, staat in software betrouwbaarheidskunde bekend onder de term 'faalintensiteit'; het aantal fouten dat optreedt per tijdseenheid. De faalintensiteit wordt veelal genoteerd als $\lambda(t)$. Op basis van uitgangspunten 1. en 2. volgt dat de faalintensiteit in het model van Jelinski en Moranda wordt gegeven door:

$$\lambda(t) = \Phi(N-(i-1))$$

Hierin is t een willekeurig tijdstip tussen het optreden van de $(i-1)$ -ste fout en de i -de fout. N is het initiële aantal fouten in de software op tijdstip $t=0$, en Φ is de evenredigheidsconstante. Bij elke fout die wordt ontdekt (en conform uitgangspunt 3. dus onmiddellijk wordt opgelost) zal $\lambda(t)$ afnemen met de evenredigheidsconstante Φ . Daardoor heeft iedere

faalintensiteit



Figuur 6; afnemende faalintensiteit

opgeloste fout een even grote impact op de afname van de faalintensiteit (zie *Figuur 6*).

2.2 Geometric model

Het Geometrische model is opgesteld door P.L. Moranda, en is een variant op het Jelinski-Moranda model en gebaseerd op de volgende uitgangspunten:

1. De mate waarin fouten optreden is constant in het tijdsinterval tussen twee opeenvolgende fouten;
2. De mate waarin fouten optreden vormt een meetkundige rij;
3. Het totaal aantal fouten in het systeem is niet naar boven begrensd;
4. De tijd tussen het optreden van twee opeenvolgende fouten volgt een exponentiële verdeling.

Uitgangspunt 2. impliceert dat de initiële waarde van de faalintensiteit $\lambda(t)$ bij het vinden van elke fout geometrisch afneemt.

Concreet: als D de waarde is van $\lambda(t)$ op tijdstip $t_0=0$, en Φ is een evenredigheidsconstante ($0 < \Phi < 1$), dan is $\lambda(t)$ na het optreden van de eerste fout gelijk aan $D\Phi$. Na het optreden van de tweede fout is $\lambda(t)$ gelijk aan $D\Phi^2$, en na het optreden van de derde fout is $\lambda(t) = D\Phi^3$.

Algemeen: na het optreden van de n -e fout in het systeem is de faalintensiteit gelijk aan: $\lambda(t) = D\Phi^n$.

De achtereenvolgende afnames van de faalintensiteit vormen als volgt een aflopende rij: $D - D\Phi, D\Phi - D\Phi^2, D\Phi^2 - D\Phi^3 \dots$ ofwel: $D(1-\Phi), D\Phi(1-\Phi), D\Phi^2(1-\Phi), \dots$

Fouten die in een vroeg stadium worden ontdekt hebben dus een grotere invloed op de reductie van de faalintensiteit dan later ontdekte fouten.

2.3 Musa Basic model

John D. Musa kan beschouwd worden als een van de grondleggers van software reliability engineering, en een fervent voorstander van het gebruik van mathematische modellen om betrouwbaarheid van software vast te stellen. Het Musa Basic model is het eerste ontwikkelde model dat gebruik maakt van de werkelijke gebruikte CPU-tijd van een systeem; alle andere tot dan toe ontwikkelde modellen maakten gebruik van werkelijk verstreken tijd (kalendertijd of "wall clock time"). De volledige naam van het model is dan ook: "*Musa's Basic execution time model*". Het model bevat overigens een component die het mogelijk maakt de resultaten op basis van tijdrekening in CPU-tijd om te zetten naar kalendertijd. Deze component relateert het gebruik van mens- en machine resources aan de CPU-tijd.

Dit model is gebaseerd op de volgende aannames:

1. Het totaal aantal fouten in het systeem is naar boven begrensd;
2. De mate waarin fouten optreden is evenredig met het aantal nog aanwezige fouten in de software;
3. De tijdsintervallen (gemeten in CPU-tijd) tussen opeenvolgende fouten zijn stuksgewijs exponentieel verdeeld.

Indien gebruik gemaakt wordt van de modelcomponent die CPU-tijd relateert aan kalendertijd, zijn er nog een aantal andere aannames die een

rol spelen. Deze aannames hebben betrekking op beschikbaarheid en inzetbaarheid van mens en machine-resources.

Musa's Basic model behoort tot de categorie "Time Between Failure" modellen. Voor het gebruik ervan zijn dus nodig de (CPU-)tijd die is verstrekt tussen de opeenvolgende opgetreden fouten, ofwel de exacte tijdstippen waarop de fouten in de software optraden. Wordt gebruik gemaakt van de component die CPU-tijd relateert aan kalendertijd, dan zijn er nog een aantal andere gegevens nodig, zoals de beschikbare resources voor het identificeren en corrigeren van optredende fouten, en de werkelijke tijd dat elke resource is ingezet.



Figuur 7; 'deur' van de Maeslantkering

3 FAALKANSANALYSE

Het uitvoeren van een faalkansanalyse bestaat uit de volgende stappen, die elk in de volgende paragrafen worden behandeld:

1. Analyse van de uitgangsdta;
2. Converteren van de beschikbare uitgangsdta tot een formaat dat toepasbaar is in het Refis/Laquso SRE-tool;
3. Selecteren van mogelijk van toepassing zijnde modellen;
4. controle op bruikbaarheid van gegevens;
5. Bepalen van uitgangspunten voor de berekening;
6. Toepassen van de modellen op de uitgangsdta en verzamelen van resultaten;
7. Controle op bruikbaarheid van resultaten (goodness-of-fit tests)
8. Interpreteren en rapporteren van resultaten.

De stappen 3 tot en met 6 worden in een aantal iteraties uitgevoerd, afhankelijk van de resultaten, om de modellen te kunnen rekaliëren op de uitgangsdta en zodoende de betrouwbaarheid van de resultaten te verhogen.

3.1 Analyse van beschikbare gegevens

De gegevens die tijdens test of exploitatie zijn verzameld over opgetreden defects vormen één deel van de invoer. Uit de registratie moeten die defects gefilterd worden die niet meegenomen worden bij de faalkansanalyse. Het gaat dan bijvoorbeeld om defects in testware, ten onrechte aangemerkte bevindingen, bevindingen in hardware (hoewel deze laatste categorie desgewenst wel meegenomen kan worden in de analyse). Een tweede deel van de invoer betreft de gebruiks- of testintensiteit (aantal uur per interval). Deze informatie komt uit de urenregistratie van het project waarbij, in geval van testtrajecten vooral naar de testuitvoering moet worden gekeken. Aangezien het geheel aan informatie soms ontbreekt of onvolledig is, is de analyse een tijdrovend onderdeel.

3.2 Conversie

3.2.1 Refis/Laquso SRE-tool



De gefilterde gegevens moeten vervolgens worden omgezet in een bestandsformaat waarin ze bruikbaar zijn voor het SRE-tool (csv, txt, dat of

sd3). Voor de TBF-varianten zijn dat de datum en tijd waarop elk defect is opgetreden. Voor FC-varianten is dat het aantal defects dat per testinterval is opgetreden en de gebruiks/testintensiteit per interval (zie ook Bijlage A). Bovendien kan bij elk defect de 'ernst' (severity) worden meegegeven.

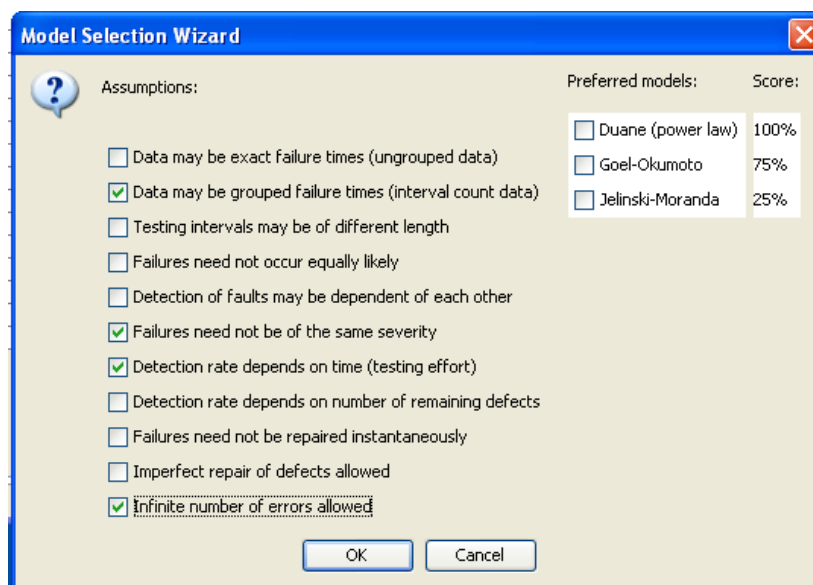
3.2.2 Aansluiting op gangbare incidentmanagement- en testtools

Als informatie over defects is vastgelegd in gangbare incidentmanagement systemen of testtools dan is het converteren daarvan naar invoer voor betrouwbaarheidsstools doorgaans geen probleem. De grootste uitdaging is gelegen in het verzamelen en converteren van bestede test- of gebruikersuren. Urenregistratie is in elke organisatie wel min of meer ingevoerd, maar de nauwkeurigheid daarvan loopt zeer uiteen. De impact van onjuiste registratie en vervolgens gebruik in betrouwbaarheidsmodellering is echter groot.

3.3 Modelselectie

Zoals in het vorige hoofdstuk is beschreven, liggen aan elk model een aantal vooronderstellingen ten grondslag met betrekking tot het systeem, de manier waarop is getest, en de eigenschappen van de defects die daarbij worden gevonden. Om vervolgens tot het juiste model (of de juiste modellen) te komen wordt de Model selectie wizard doorlopen:

Figuur 8 toont de modelselectie wizard uit het SRE-tool met daarin de belangrijkste onderscheidende kenmerken, voorwaarden of aannames. Wanneer een voorwaarde of aanname van toepassing is wordt deze aangevinkt en berekend de wizard de mate waarin de verschillende groeimodellen hierop aansluiten.



Figuur 8; modelselectie wizard

3.4 Bruikbaarheid van gegevens

Na conversie van de aangeleverde gegevens dient te worden nagegaan of en in hoeverre deze geschikt zijn voor een statistische analyse. Aangezien

alle betrouwbaarheidsanalyses als uitgangspunt hebben dat de ingevoerde defectgegevens op den duur een betrouwbaarheidsgroei laten zien, is het dus allereerst zaak vast te stellen of de ruwe data inderdaad een betrouwbaarheidstoename vertoont, bijvoorbeeld m.b.v. het voortschrijdend gemiddelde van het aantal defects per tijdseenheid (faalintensiteit).

3.4.1 Voortschrijdend gemiddelde

Een veel gebruikte methode hiervoor is die van het 'voortschrijdende gemiddelde'. Voor de "Time Between Failures" (TBF) variant is het voortschrijdend gemiddelde G_i na het i -de defect gedefinieerd als:

$$G_i = \frac{\sum_{j=1}^i T_j}{i}$$

Hierbij is T_j de tijd tussen het $(j-1)$ -ste en j -de defect.

Indien het voortschrijdend gemiddelde over een serie defectgegevens in TBF-formaat een stijgende lijn vertoont, is er sprake van een stijgende trend in de time between failures. Derhalve is sprake van een groei in de betrouwbaarheid van het systeem.

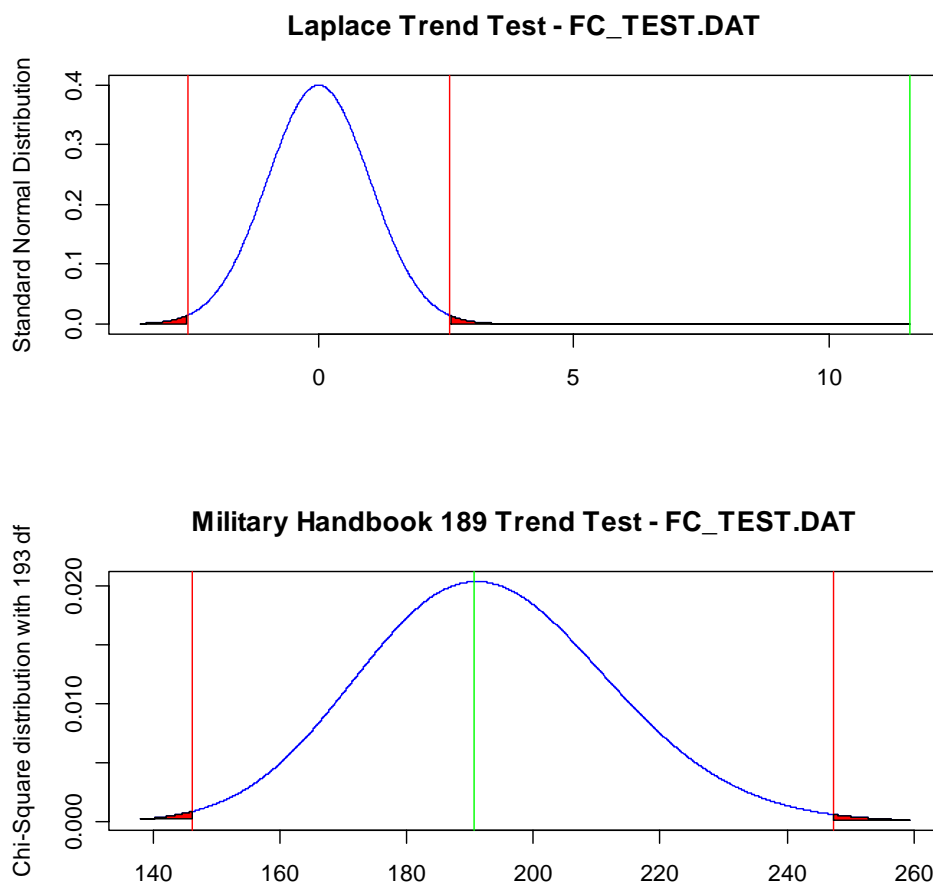
Zijn de gegevens beschikbaar in de vorm van aantallen defects per testinterval ("Failure Count" - FC) dan is het voortschrijdend gemiddelde G_i na het i -de testinterval:

$$G_i = \frac{\sum_{j=1}^i N_j}{i}$$

N_j is het aantal defects dat is opgetreden in het j -de testinterval.

Indien het voortschrijdend gemiddelde over een serie defectgegevens in FC-formaat een dalende lijn vertoont is er sprake van een dalende trend in het aantal fouten per testinterval. Derhalve is sprake van een groei in de betrouwbaarheid van het systeem. In bijgaande *Figuur 9* is voor een twintigtal testintervallen het aantal gevonden defects per testinterval weergegeven, samen met het voortschrijdend gemiddelde over deze defects. Uit de ontwikkeling van deze laatste valt af te leiden dat er vanaf testinterval 7 sprake is van een dalende trend.

Het Refis/Laquso SRE-tool biedt bovendien geavanceerde trendanalyses als de Laplace test en de Military Handbook 189 test (zie *Figuur 9*)



Figuur 9; Trendanalyses in Refis/Laquso SRE-tool

3.5 Uitgangspunten voor berekening

Voordat de modellen kunnen worden toegepast maakt de analist bepaalde keuzes over de toe te passen methode van schatting, al of niet filteren van gegevens en andere parameterinstellingen, deels afhankelijk van de toe te passen modellen. De meest elementaire parameters worden hieronder toegelicht:

- Parameter schatting methode
- Gegevensbereik (al of niet modelspecifiek)
- Filters (functioneel en statistisch)
- Voorspelling

3.5.1 Parameter schatting

Parameter schattingen van de modellen ten opzichte van de werkelijke data kunnen op twee manieren berekend worden: volgens de maximale waarschijnlijkheidsfunctie (maximum likelihood) of kleinste-kwadraat methode (least-square method). Zonder in detail te treden geldt dat de eerste de meest betrouwbare resultaten geeft, doch niet altijd toepasbaar is. De tweede is weliswaar altijd toepasbaar, doch blijkt minder betrouwbaar. Gangbaar is derhalve eerste maximum likelihood toe te passen en afhankelijk van de resultaten daarvan desgewenst de berekening opnieuw uit te voeren met de least squares methode.

3.5.2 Gegevensbereik

Mede op basis van de uitkomsten van bruikbaarheidscontrole bepaalt de analist welk deel van de uitgangsddata als invoer voor de modellen in aanmerking komt. In de rapportage wordt hierover verslag gedaan.

3.5.3 Filters

Met behulp van een filter kan het gegevensbereik eveneens beïnvloed worden door defectgegevens van een bepaalde categorie buiten beschouwing te laten, of juist mee te nemen. Categorieën kunnen zijn samengesteld op basis van ernst, prioriteit of aard van de gevonden defects.

Daarnaast kan het om statistische redenen noodzakelijk zijn bepaalde gegevens uit de uitgangsgegevens te filteren. Defectgegevens die afkomstig zijn uit onbetrouwbare waarnemingen, buitenproportionele verstoringen in een trend, afronding van geconverteerde gegevens, etc. Waar van toepassing wordt daarvan in de rapportage melding gemaakt.

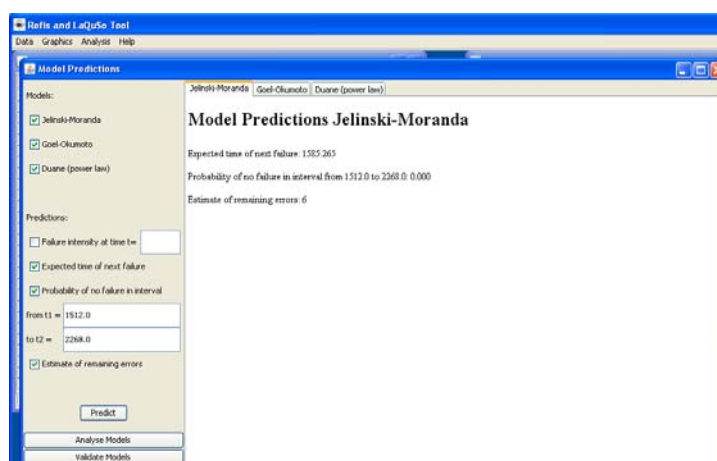
3.5.4 Voorspelling

Tot slot geeft de analist aan voor hoeveel defects of testintervallen (voor resp. TBF en FC modellen) de modellen een voorspelling moeten afgeven. Deze parameter wordt ingesteld op basis van de door de opdrachtgever geformuleerde eisen.

3.6 Resultaten

De resultaten van het toepassen van één of meerdere geselecteerde modellen op de uitgangsddata, worden in principe weergegeven op dezelfde manier als waarop de eisen aan de betrouwbaarheid zijn geformuleerd. Dat kan bijvoorbeeld op de volgende wijzen:

1. Betrouwbaarheid / Faalkans
2. Cumulatief aantal defects
3. Foutintensiteit



Figuur 10; Model voorspellingen

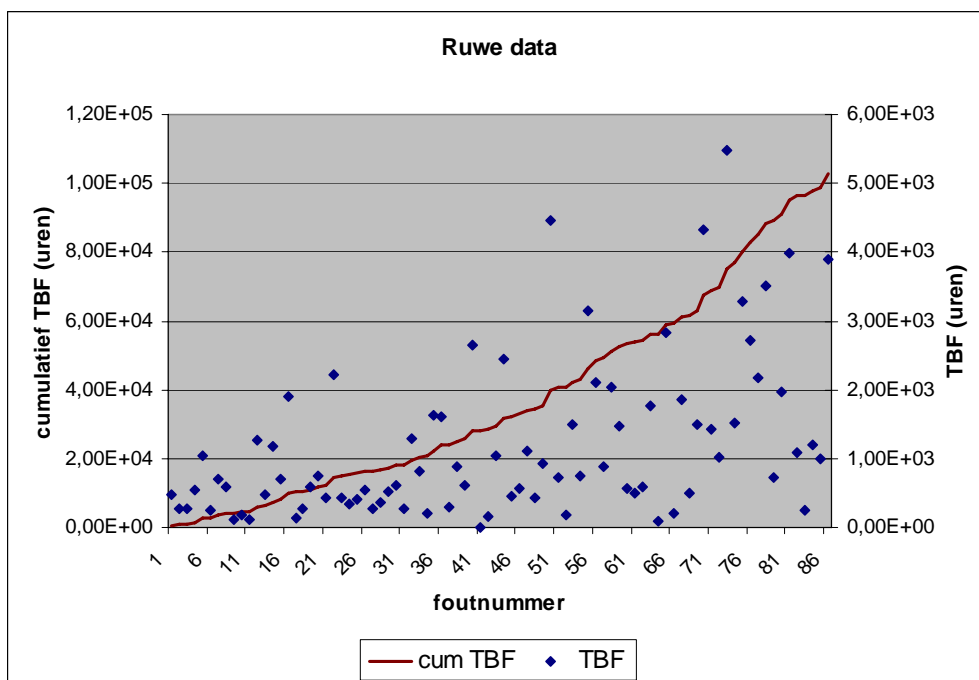
Bij de resultaten hoort tevens een analyse van de uitgangsgegevens (w.o. controle op bruikbaarheid, zie § 3.4) een verklaring van de Goodness-of-fit van de gebruikte modellen en een beschrijving van het eventuele be-

trouwbaarheidsinterval waarbinnen de werkelijke waarden zich volgens een bepaald betrouwbaarheidsniveau zullen bevinden.

De hierna getoonde voorbeelden zijn afkomstig van een fictieve case.

3.6.1 *Ingangsdata*

De gegevens die zijn gebruikt om de hierna genoemde resultaten te tonen, zijn afkomstig van een voorbeeld defectregistratiesysteem. *Figuur 11* toont deze Time Between Failures (TBF) voor elk defect (x-as) in (blauwe) punten uitgezet tegen de secundaire y-as (rechts). De cumulatieve TBF is uitgezet in een lijn tegen de primaire y-as (links).

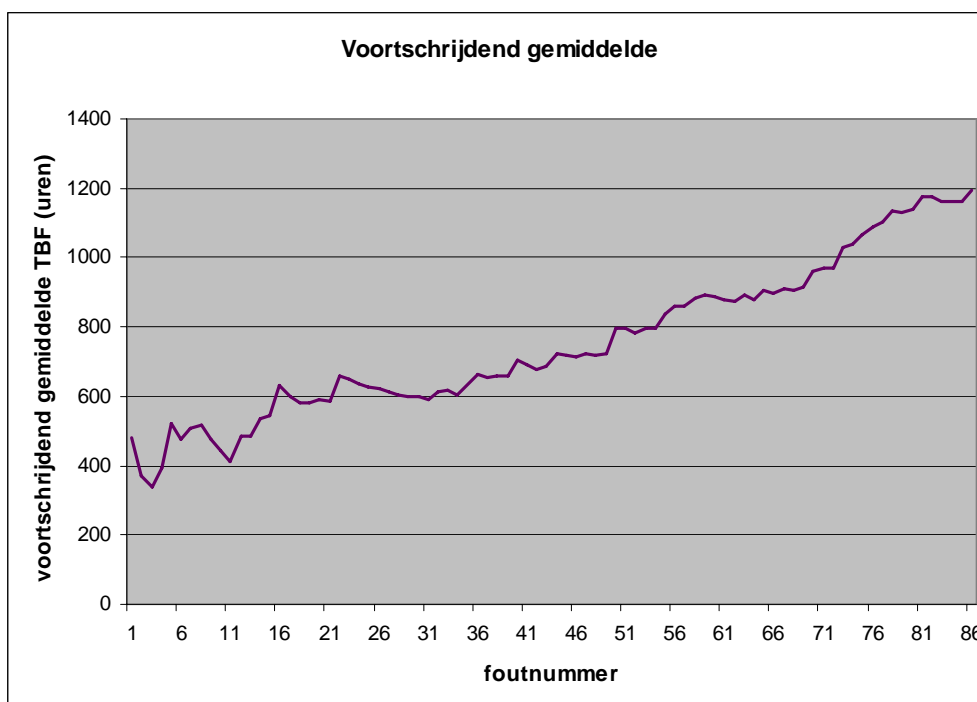


Figuur 11; ruwe data

In 102594 uur testen zijn in totaal 86 defects gevonden van gelijke ernst. Van elk defect is de TBF t.o.v. vorige defect vastgelegd.

3.6.2 Voortschrijdend gemiddelde

De eerste controle op betrouwbaarheidstoename bestaat uit het bepalen van het voortschrijdend gemiddelde op de TBF (§ 3.4.1¹). *Figuur 12* toont de bij deze case behorende grafiek. Er is sprake van een betrouwbaarheidsgroei, die vanaf foutnummer 31 ook redelijk constant is.



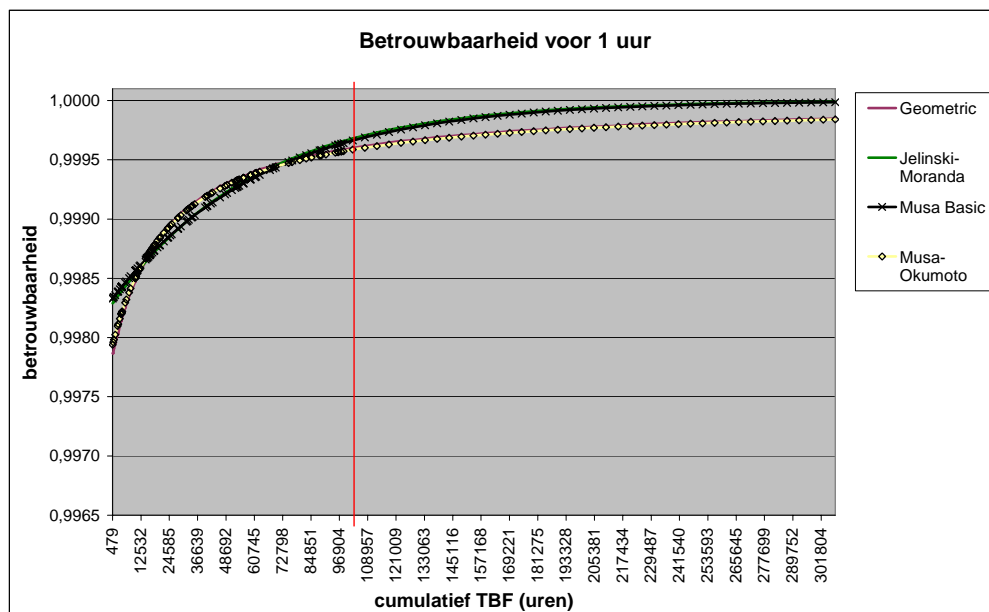
Figuur 12; voortschrijdend gemiddelde TBF in uren

In deze case is gekozen voor het toepassen van 4 basis modellen, voor de volledige reeks van fouten. De volgende paragrafen tonen de uitkomsten daarvan.

¹ Merk op dat in deze paragraaf sprake is van een grafiek voor voortschrijdend gemiddelde van Time Between Failure en in paragraaf 3.4.1 een voorbeeld grafiek van een andere case wordt getoond waarin sprake is van voortschrijdend gemiddelde van aantal defects per interval (Failure Count). Zie ook Bijlage A.

3.6.3 *Betrouwbaarheid / Faalkans*

De betrouwbaarheid van een systeem kan geformuleerd worden voor willekeurige tijdseenheden (zie § 1.2.1). *Figuur 13* toont tot de verticale (rode) lijn de betrouwbaarheidsgroei over de doorlopen 102594 testuren, en vanaf daar, de voorspelling voor het komende (test- of gebruiks-)uur.



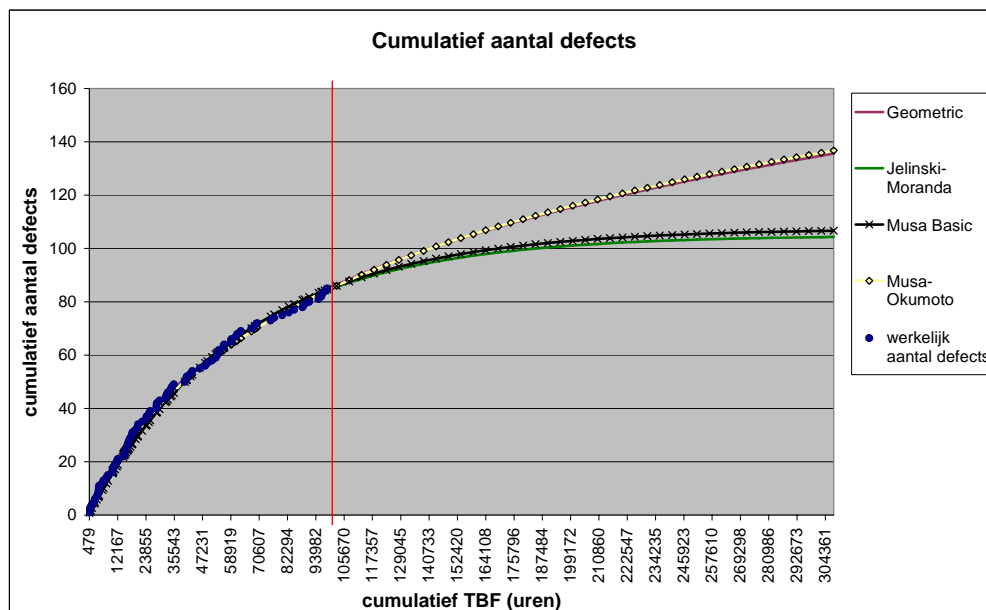
Figuur 13; betrouwbaarheid voor 1 uur

Daaruit valt af te leiden dat de betrouwbaarheid van de software van het systeem in kwestie na voltooiing van de tests, iets boven de 0,9995 ligt voor een periode van 1 uur. Bij de grafiek hoort een overzicht van de berekende waarden waaruit de precieze getallen blijken.

Met andere woorden; de kans dat de software in het eerstvolgende uur bij een gelijksoortig gebruik als tijdens de uitgevoerde tests, een fout vertoont is ongeveer 0,0005 (1 minus betrouwbaarheid).

3.6.4 *Cumulatief aantal defects*

Op basis van de berekende betrouwbaarheid, voorspellen de modellen nog op te treden aantallen defects voor volgende of test- en gebruiksuren. In *Figuur 14* is het cumulatief aantal defects getoond dat door de toegepaste modellen wordt voorspeld. Uit de bijgeleverde cijfers blijkt dat het totaal aantal defects in de software zal oplopen tot tussen de 104 (JM) en 136 (MO).

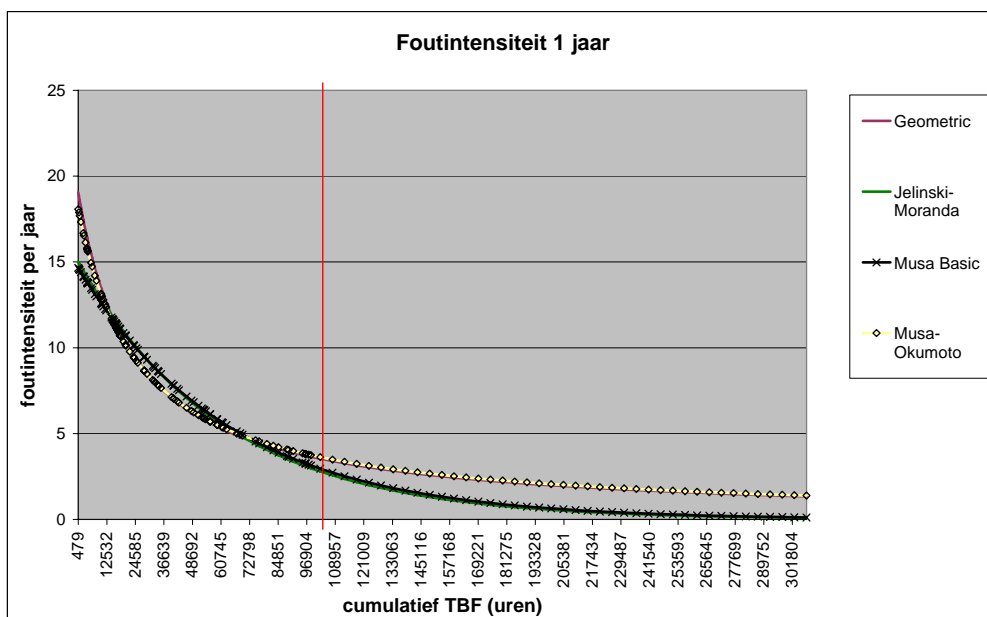


Figuur 14; cumulatief aantal defects

De werkelijk gemeten defects, 86 in totaal, zijn weergegeven in de (blauwe) markers en de door de modellen voorspelde waarden in de krommen. Afhankelijk van het van toepassing zijnde model mag men in totaal dus nog 18 tot 50 nieuwe defects verwachten na ingebruikname van het systeem.

3.6.5 *Foutintensiteit*

Foutintensiteit is het aantal defects per test- of gebruiksduur. *Figuur 15* toont de afname van het aantal defects per jaar. Ook hier is met de rode (verticale) lijn aangegeven waar de werkelijke (basis)gegevens ophouden en de voorspelling op grond van toegepaste modellen begint. Vóór deze lijn is de foutintensiteit gebaseerd op de werkelijk Time-Between-Failure (TBF), daarna is het een voorspelling voor de toekomst. De intensiteit is in lijnen uitgezet tegen de y-as.



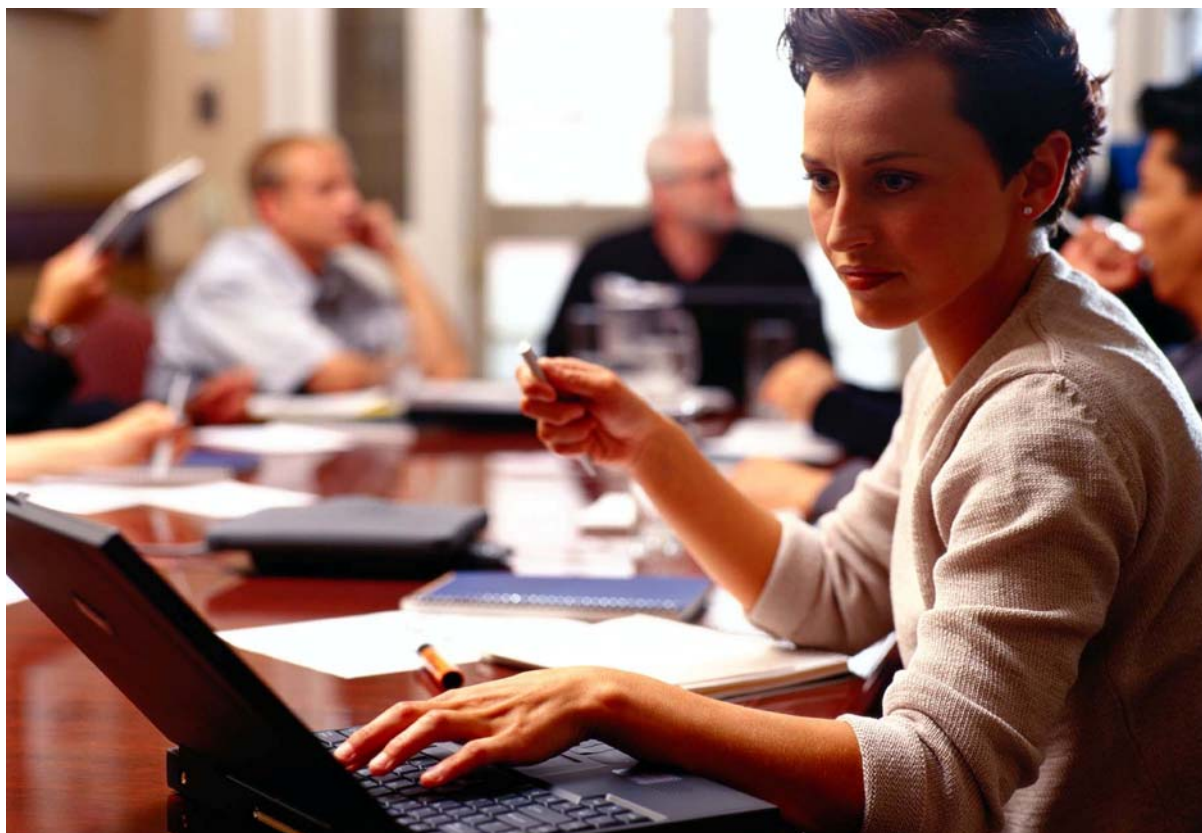
Figuur 15; foutintensiteit per jaar

Zoals uit de bijgeleverde cijfers is af te lezen, bedraagt de foutintensiteit voor het eerstvolgende test- of gebruiksjaar tussen de 3,7 en 2,6.

Afhankelijk van het systeem en de manier waarop de eis is geformuleerd, kan de foutintensiteit worden weergegeven per seconde, minuut, uur, dag, week, maand of jaar.

3.6.6 *Statistische significantie*

In de Refis/Laquso SRE-tool wordt gebruik gemaakt van twee soorten Goodness-of-fit tests: Zhao-Wang (voor NHPP modellen zoals Goel-Okumoto) en Kolmogorov. Bij de analyse wordt bepaald of en in hoeverre de gebruikte modellen voldoen aan de grenswaarden voor deze tests (ofwel of de hypothese dat het model past op de aangeboden gegevens verworpen kan worden).



4 SAMENVATTING

4.1 Softwarebetrouwbaarheid

Betrouwbaarheid van software wordt uitgedrukt in een getal tussen 0 en 1, zijnde de kans dat de software functioneert conform specificaties voor een bepaalde periode in tijd onder bepaalde omstandigheden. Betrouwbaarheidsmodellen zijn gebaseerd op het uitgangspunt dat de betrouwbaarheid van software toeneemt na elk gevonden en verholpen defect.

In de loop van de afgelopen 30 jaar zijn een groot aantal van dergelijke modellen ontwikkeld. Deze modellen kunnen op diverse manieren worden gecategoriseerd.

Het overgrote deel van de modellen is gebaseerd op klassieke statistische theorie. In het Refis/Laquso SRE-tool zijn echter ook Bayesiaanse modellen opgenomen (zoals het model van Littlewood en Verall) [BRA]. Verder onderscheiden de modellen zich voornamelijk door de aannames over de vorm van de faalintensiteitsfunctie, en de onderliggende kansverdeling van de tijd tussen opeenvolgende optredende fouten c.q. het aantal fouten per tijdsinterval.

4.2 Faalkansanalyse

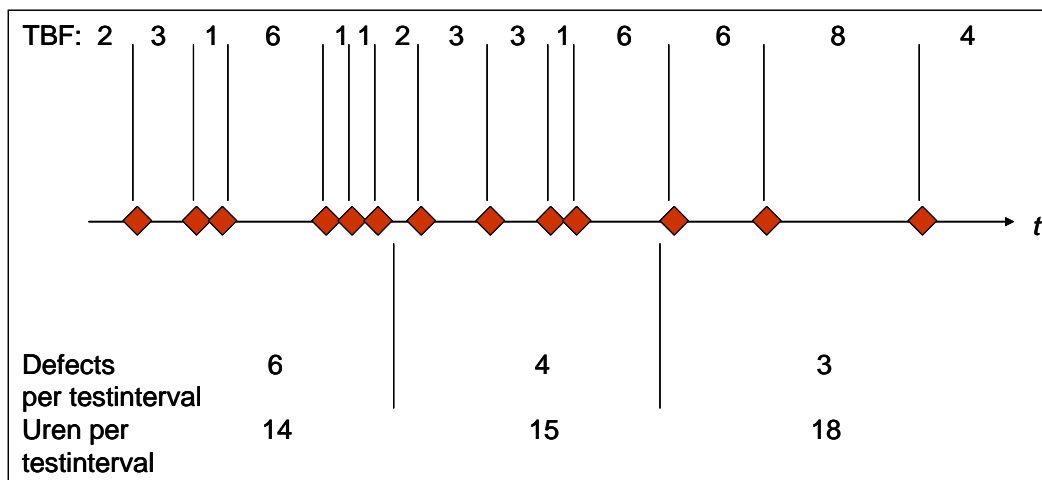
Het uitvoeren van een Faalkansanalyse bestaat uit de volgende stappen:

1. Analyseren van de beschikbare uitgangsdta (bevindingen, defect- en urenregistratie);
2. Converteren van de beschikbare uitgangsdta tot een formaat dat toepasbaar is in het SRE-tool;
3. Selecteren van één of meerdere mogelijk van toepassing zijnde modellen, op basis van de kenmerken van het systeem, de manier waarop het wordt gebruikt (of getest) en de defects worden opgelost;
4. Controle op bruikbaarheid van de uitgangsdta, onder meer met behulp van voortschrijdend gemiddelde en Laplace Test;
5. Bepalen van uitgangspunten voor de berekening; onder meer "maximum likelihood" versus "least squares method";
6. Toepassen van de modellen op de uitgangsdta en verzamelen van resultaten. Resultaten kunnen voor verschillende periodes in tijd en eventueel uitgesplitst naar ernst of aard, worden weergegeven als:
 - a. Betrouwbaarheid/ faalkans
 - b. Cumulatief aantal defects
 - c. foutintensiteit
7. Controle op bruikbaarheid van resultaten met behulp van Goodness-of-fit tests;
8. Interpreteren en rapporteren van resultaten; samenstellen van een eindrapport of het opnemen van de gegevens in een management dashboard.

Bijlage A TIME BETWEEN FAILURE VERSUS FAILURE COUNT PER INTERVAL

Betrouwbaarheidsmodellen berekenen de betrouwbaarheidsgroei op basis van het aantal defects dat is gevonden en opgelost tijdens testen of gebruik van het system. Daarbij kan onderscheid worden gemaakt tussen de werkelijke tijd tussen het optreden van twee fouten (Time Between Failure - TBF) en het aantal fouten per test- of gebruiksinterval (Failure Count - FC). Een interval is in dat geval een periode in tijd waarin het systeem getest of gebruikt wordt en eventuele defects worden verzameld om aan het eind van het interval te worden opgelost. Verschillende intervallen kunnen van verschillende lngte of intensiteit zijn. En beide varianten kunnen in het SRE-tool worden ingevoerd.

Figuur 16 toont met een voorbeeld het onderlinge verband tussen deze twee vormen van uitgangsdta. Op een tijdslijn (t) zijn de defects die tijdens het testen zijn gevonden met rode markers aangegeven. Daarboven is de Time Between Failure (TBF) weergegeven in uren (incl. tijd vanaf begin van testen en tijd tot het einde van testen). Daaronder is aangegeven hoe de defects verdeeld zouden zijn over de drie fictieve testintervallen die in dit traject zijn onderkend. In testinterval 1 zijn 6 defects gevonden. De testtijd in het interval bedraagt 14 uur (alle TBF's opgeteld).



Figuur 16; Time Between Failure versus Failure Count

Aan de hand van dit voorbeeld is meteen duidelijk dat TBF-modellen nauwkeuriger zijn dan FC-modellen. Op basis van de testinterval gegevens alleen is immers niet bekend op welk moment binnen het interval defects optraden. Ook wordt duidelijk dat TBF gegevens omgezet kunnen worden naar FC gegevens, maar niet andersom, althans niet zonder aannames (in sommige gevallen is het mogelijk FC gegevens te converteren naar TBF bijvoorbeeld door aan te nemen dat de defects gelijkmatig over een testinterval zijn gespreid).

INDEX

bestandsformaat, 16	Geometric modell, 12
betrouwbaarheid, 5	Jelinski-Moranda, 11
betrouwbaarheidscurven, 6	John D. Musa, 12
betrouwbaarheidsgroei, 7	least-square method, 18
betrouwbaarheidsgroei-curves, 8	maximum likelihood, 18
Capability Maturity Model, 5	Musa Basic model, 12
faalintensiteit, 11	voorspelling, 19
filter, 19	voortschrijdende gemiddelde, 17

FIGUREN

<i>Figuur 1; betrouwbaarheidscurve</i>	6
<i>Figuur 2; betrouwbaarheidscurves na oplossen van 7 individuele defects</i>	7
<i>Figuur 3; betrouwbaarheidsgroei voor 4 periodes</i>	7
<i>Figuur 4; betrouwbaarheidsgroei-curven</i>	8
<i>Figuur 5; betrouwbaarheidsgroei-curven logaritmisch</i>	9
<i>Figuur 6; afnemende faalintensiteit</i>	11
<i>Figuur 7; 'deur' van de Maeslantkering</i>	13
<i>Figuur 8; modelselectie wizard</i>	16
<i>Figuur 9; Trendanalyses in Refis/Laqueso SRE-tool</i>	18
<i>Figuur 10; Model voorspellingen</i>	19
<i>Figuur 11; ruwe data</i>	20
<i>Figuur 12; voortschrijdend gemiddelde TBF in uren</i>	21
<i>Figuur 13; betrouwbaarheid voor 1 uur</i>	22
<i>Figuur 14; cumulatief aantal defects</i>	23
<i>Figuur 15; foutintensiteit per jaar</i>	24
<i>Figuur 16; Time Between Failure versus Failure Count</i>	29

LITERATUUR

- [BRA] Ed Brandt, Rob Henzen, Isaac Corro Ramos and Alessandro di Bucchianico "Two Case Studies in Applying Statistical Models in Software Development", Quality Engineering, 19:4 (2007), 339 - 351
- [CAL] D. Baker c.s. "Computer Aided Software Reliability Estimation", University of Calgary, feb. 2002; Zie: http://www.ucalgary.ca/~dbmenks/seng/seng609.11/casre.html#tool_comparison
- [HUM] W.S. Humphrey, "Managing the Software Process", Addison-Wesley Publishing company, 1989
- [LYU] M.R. Lyu, "Handbook of Software Reliability Engineering", McGraw Hill, New York, 1996
- [MUS1] J.D. Musa, A. Iannino, K. Okumoto, "Software Reliability: Measurement, Prediction, Application", McGraw Hill, New York, 1987

- [NIK] Nikora, A. P. "CASRE User Guide", Version 3.0. Autonomy and Control Section, Jet Propulsion Laboratory. NASA, 2000.
- [ZEI] B. van Zeist, P. Hendriks, R. Paulussen, J. Trienekens "Kwaliteit van software producten", Kluwer Bedrijfswetenschappen, 1996