Transactions of KSAE, Vol. 19, No. 4, pp.1-7 (2011)

1225-6382/2011/112-01

차량용 임베디드 소프트웨어 신뢰성평가 연구

백 재 진*

부퍼탈대학교 안전공학과

A Study on Reliability Evaluation of Embedded Software in Vehicle

Jaejin Baek*

Department of Safety Engineering, University of Wuppertal, Gaussstrasse 20, 42119 Wuppertal, Germany
(Received 7 October 2009 / Accepted 20 March 2011)

Abstract : Various convenient systems which are telecommunication and navigation system and safety system which include Antilock Braking System, Electronic Stability Control, Adaptive Cruise Control have been developed and applied to meet customer needs and each standards since vehicles are used. The complexity of new electronics become significant reason of breakdown especially embedded software failures. Hardware reliability is almost stabilized with long history but software reliability needs more improvements through reliability researches. This new challenge will improve software reliability to clear its overall failures in vehicles. This paper introduces some software reliability models and evaluates embedded software reliability using failure data which occurred during operating.

Key words : Embedded software(임베디드소프트웨어), Software reliability(소프트웨어신뢰성), Software reliability model(소프트웨어신뢰성모델), Failure data(고장자료)

1. 서 론

기계부품 중심으로 구성되어 있던 차량이 전기/전자기술의 발전, 다양하고 새로운 요구사항, 법적 규제로 인하여 전기/전자부품 중심으로 구성되고 있으며 이러한 변화는 2003년 전기에 의해 작동되는 부품의 비율이 20%에서 2015년 40%로 증가할 것으로 보고하고 있는 McKinsey[1)]의 연구를 통해서도 알 수 있다.

사용자의 편의성 증대를 위한 시스템(Telecommunications system, Navigation system), 안전관련 시스템(Antilock Braking System , Electronic Stability Control, Adaptive Cruise Control)과 같은 전기/전자부품의 사용은 다양한 요구사항을 충족시키고, 사고를 예방하며 사고 시 인명피해를 최소화 하는 장점이 있지만 이를 구현하기 위한 차량통신시스템, 소프트웨어 라인 수, 소프트웨어로 구현되는 기능수의 증가로 인하여 차량이 복잡해진다는 단점이 있다. 이러한 문제는 비단 개발자 뿐 아니라 품질, 안전, 신뢰성전문가 역시 새롭게 도전해야 할 과제로 대두되고 있으며 특히, 전기/전자부품의 개발은 하드웨어와 소프트웨어를 고려하여 동시에 개발이 이루어지기에 요구사항 분석에서 부터 개발완료까지 많은 노력이 필요하다.

자동차분야에서는 이러한 문제점을 해결하기 위해서 표준을 사용하거나 기업 환경에 적합한 개발방법론을 개발하고 있다. 예컨대, IEC 61508, ISO 26262는 E/E(Electrical/Electronic)하드웨어와 소프트웨어와 관련된 개발 전 과정을 기술하여 제품의 신뢰성향상에 도움이 되고 있으며 V-Model의 경우 표준은 아니지만 하드웨어, 소프트웨어의 요구분석

*Corresponding author, E-mail: jaejin.baek@gmail.com

에서부터 양산까지 모든 사항을 고려 할 수 있기 때문에 자동차산업분야에서 E/E 개발 시 빈번하게 사용되는 개발방법론이다.

하지만 이러한 표준과 개발방법론에도 불구하고 차량의 오류발생원인 중 가장 큰 부분은 E/E 분야이며 내장되어 있는 소프트웨어에 의한 고장이 증가하고 있으며 이는 1998년에서 2001년 사이에 차량 고장의 원인 소프트웨어가 아닌 다른 고장원인은 3%만 증가한 반면 소프트웨어에 의한 고장은 23%나 증가한 사실을 통해서 차량고장의 원인 중 소프트웨어의 비중이 증가하고 있음을 알 수 있다.[2)]

본 논문에서는 자동차분야에서 전기/전자부품개발을 위한 V-Model을 간략하게 소개하고 차량용 내장소프트웨어에 대한 전반적인 특성소개와 함께 내장되어 있는 소프트웨어 신뢰성을 정량적으로 평가하기 위한 소프트웨어 신뢰성모델을 소개한다. 소개되는 모델의 경우 테스트기간 중 발생한 고장을 이용하여 신뢰성모델링을 위해 사용되지만 본 논문에서는 실제 운영 중 발생한 고장자료를 이용하여 신뢰성모델링의 적합여부를 관찰하려고 한다.

2. 임베디드 소프트웨어

임베디드 소프트웨어는 차량용 제어장치, 로봇, 텔레비전, 의료기기 등과 같은 전자장치에 내장되어 있는 소프트웨어를 의미한다. 특히, 차량용 임베디드 소프트웨어는 조향시스템, 제어시스템과 같이 실시간으로 작동해야하는 시스템의 제어에 많이 사용이 된다. 이러한 소프트웨어는 정확한 값을 계산하는 것 뿐 아니라 시간의 지연 없이 일을 처리해야 한다. 이처럼, 복잡한 요구사항들과 안전 Critical 한 기능을 담당하는 소프트웨어의 신뢰성은 더 중요시되고 있다.

하드웨어의 경우 고장 발생에 대한 사전 징후 예컨대, 심한진동, 열화 등을 확인할 수 있어 고장을 예방할 수 있으나 소프트웨어의 경우 사전에 어떠한 경고 없이 발생하기에 물리적으로 이를 예방하기는 어려운 실정이다.[3)] 또한 혼용되어 사용되어지는 Error, Fault, Failure의 정의 때문에 관찰대상을 정확하게 인지 못하는 어려움이 있기에 에러, 결함, 고장의 정의를 살펴보았다.[4)]

- 에러(Error): 인간의 행동이 잘못된 결과를 발생시키는 의미로 예컨대, 프로그래머 또는 운전자의 잘못된 행동
- 결함(Fault): 잘못된 경과, 과정 또는 컴퓨터 프로그램에서 잘못된 자료정의
- 고장(Failure): 시스템 또는 구성요소가 정의된 작업 요구사항 내에서 요구되는 기능을 수행하지 못함

위의 정의로부터 본 논문에서는 MOST BUS기반 Bluetooth Connectivity 시스템을 관찰하였으며 내장된 소프트웨어에 의해 시스템이 요구사항에 맞게 작동되지 않는 경우를 고장으로 정의하였다. 시스템에 에러나 결함이 내포되어 있어도 해당 기능이 실행되지 않을 경우 고장이 발생하지 않을 수 있기에 본 논문에서는 에러나 결함이 아닌 고장자료를 이용하여 소프트웨어의 신뢰성을 평가하였다.

또한 임베디드 소프트웨어는 다음과 같은 문제점을 지적할 수 있다. 일반 소프트웨어 예컨대, OS, Internet 프로그램은 일반적으로 PC에서 개발되어 PC에서 사용되어 개발과 사용이 동일한 플랫폼에서 이루어지나 임베디드 소프트웨어는 일반적으로 PC에서 개발되어 다른 플랫폼 예컨대, 핸드폰, 차량용 제어장치 등에서 사용되어 개발과 사용 플랫폼이 서로 다른 문제점이 있다.

이러한 문제점은 임베디드 소프트웨어 개발에 대한 문제점과 소프트웨어가 고장발생 원인 중 큰 비중을 차지하게 된다는 문제점을 가지게 되며 임베디드 소프트웨어는 하드웨어와 동시에 개발이 이루어지며 서로 다른 과정을 통해 개발되어 추후에는 통합되어야 하는 어려움도 내포하고 있다.

이러한 문제점을 해결하기 위해서 소프트웨어의 모든 수명 기를 관찰할 수 있는 방법론을 사용하며 테스트기간 정책 역시 하나의 방법으로 사용하고 있다.

Fig. 1는 테스트기간이 길어지면 더 많은 고장을 테스트기간동안 수정하게 되어 Release되면 고장발생을 줄일 수 있지만 테스트비용과 납기일이 늦어진다는 문제가 있으면 테스트기간이 짧게 되면 상대적으로 적은 고장을 테스트기간동안 수정하게 되어 Release 후 발생하는 고장을 수정하기 위해 많은

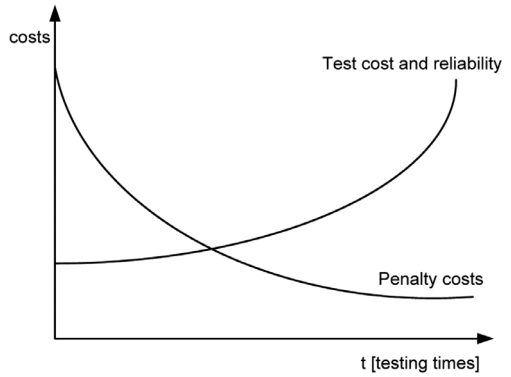

Fig. 1 Costs vs. testing time

비용과 기업 이미지손상에 큰 영향을 주게 된다. 이러한 많은 노력에도 불과하고 운영 중에는 예상치 못한 오류가 발생하는 문제점을 가지고 있다.

3. V-Model

오늘날 자동차산업분야에서는 생산되는 제품의 품질 및 신뢰성향상을 위해 국제표준이나 기업 환경에 맞게 개발된 방법론을 사용한다. 이는 개발과정의 관리를 통해 생산되는 제품의 품질 및 신뢰성이 향상된다는 관점으로 사용하고 있으며 또 다른 방법으로는 FMEA, Fault Avoidance Technique, Fault Tolerant Technique[5]과 같이 직접적인 소프트웨어의 품질 및 신뢰성향상을 위한 방법이 사용된다. 특히 자동차의 전기/전자 시스템 개발 시에는 하드웨어와 소프트웨어 개발을 동시에 고려할 수 있는 V-Model을 많이 활용되고 있으며 이 V-Model 개발되는 시스템에 따라 테일러링 되어 사용될 수 있다.

본 장에서는 관찰된 시스템 개발에 사용된 V-Model에 대해서 일반적인 개발절차에 대해서 소개하려고 한다. 이 모델은 폭포수개발방법론에 근거하여 1990년 대 초 독일에서 개발된 방법론으로 크게 검증(Verification)과 확인(Validation)의 과정으로 나눠진다.

검증부분에서는 개발되는 시스템의 요구사양이 분석되며 논리적 시스템구조가 명세화 된다. 논리적 시스템구성이 실제로 구현이 가능한지를 확인하고 어떠한 부분을 하드웨어로 구현할지 소프트웨어로 구현할지 결정한다. 소프트웨어 구현을 위한 요구사항이 분석되며 소프트웨어간의 상호작용, 소프트웨어와 하드웨어간의 상호작용을 기술하며 확인단계의 마지막을 거친 후 개발자는 소프트웨어 코드를 작성하게 된다. 검증단계를 통하여 얻어지는 문서는 모든 개발과정에서 사용되기에 명확한 요구사양 도출과 문서화가 반드시 이루어져야 한다.

확인단계의 시작은 소프트웨어 모듈 테스트로 일반적으로 화이트박스 테스트방법을 이용하여 개발자에 의해 테스트된다. 모듈 테스트가 완료되면 모듈들이 서로 통합되어 개발자가 아닌 테스트전문가에 의해 블랙박스 테스트방법을 통해 수행된다. 테스트 완료 후 통합 소프트웨어가 하드웨어에 내장되며 일반적으로 자동화 테스트도구를 통하여 테스트된다. 시스템 요구사양을 이용하여 최종 테스트가 이루어진다.

4. 소프트웨어 신뢰성모델

4.1 고장형상

소프트웨어가 차량용 Electronics에 내장되어 운영 중 발생한 고장자료를 이용하여 관찰대상이 고장에 소요되는 시간을 살펴 본 결과 Fig. 2와 같이 나타낼 수 있다.

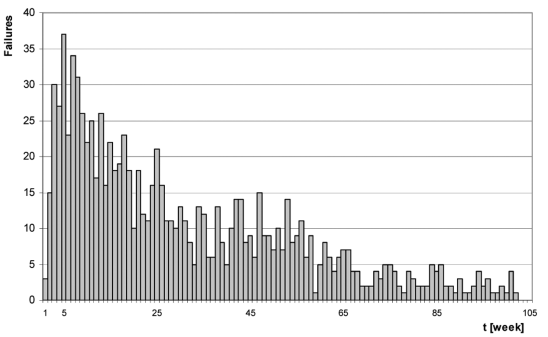

Fig. 2 Observed failure number per unit time t[week]

4.2 고장자료

소프트웨어 신뢰성모델을 사용하여 신뢰성평가를 하기 위해서는 고장발생 정보가 필요하다. 운영 중 고장이 발생하면 이 부품은 다시 고장발생원인에 대한 분석에 들어가고 이 결과 하드웨어 고장인지 소프트웨어 고장인지 구분되어 지고 여기에서

Table 1 Failure database

Pro_date	Reg_date	Failure_date
2001-01-15	2001-02-01	2002-10-25
2001-01-15	2001-02-01	2003-10-25
2001-05-01	2001-07-07	2001-05-25
2001-05-01	2001-08-15	2003-01-25

신뢰성평가에 필요한 실제 고장이 발생하기 까지 소요된 시간을 구할 수 있게 된다.

Table 1은 소프트웨어 신뢰성평가에 필요한 최소한의 자료구조로 이 이외의 소프트웨어 버전, 고장모드 등의 필요사항에 따라서 고장자료를 구성할 수 있으며 “Pro_date”는 임베디드 소프트웨어가 전기/전자제품에 내장되어 제품화 된 시점을 말하면 “Reg_date”는 실제 부품이 조립되어 운영이 시작된 일자를 의미하며 “Failure_date”는 운영 중 고장이 발생한 시간을 의미한다. 따라서 소프트웨어의 고장시간은 고장원인이 소프트웨어에 의한 것으로 고장발생시간과 운영시작일의 차이를 의미한다.

4.3 소프트웨어 신뢰성 모델

4.3.1 Goel-Okumoto 모델

Fig. 2와 같이 소프트웨어 고장현상은 단위시간당 고장횟수의 형태로 그 고장현상을 기술할 수 있으며 이는 Nonhomogeneous Poisson Process(NHPP) 기반 확률론적 신뢰성모델을 통해 임베디드 소프트웨어 신뢰성을 모델링할 수 있다.

즉, NHPP 소프트웨어 신뢰성 모델은 관측시간 (0, t] 사이에서 관찰된 고장수 $m(t)$를 모델링하기 위해서 사용되며 1979년 Amrit Goel와 Kazu Okumoto에 의해서 처음으로 제안되었으며 식 (1)과 같이 나타낸다.[6)]

$$m(t) = n_0(1-e^{-\lambda t}), \quad (1)$$
$$n_0(t) = n_0,$$
$$\lambda(t) = \lambda.$$

where

λ: is failure detection rate,

$m(t)$: the number of failure detected up to the time of observation,

n_0 : initial failure number.

4.3.2 Inflection S-shaped 모델

1984년 Ohba에 처음 제안되었으며 소프트웨어 Failure 감지 프로세스를 분석하기 위해서 개발되었으며 식 (2)와 같이 나타낸다.[7)]

$$m(t) = \frac{n_0(1-e^{-\lambda t})}{1+\beta e^{-\lambda t}}, \quad (2)$$
$$n_0(t) = n_0,$$
$$\lambda(t) = \frac{\lambda}{1+\beta e^{-\lambda t}}.$$

where

λ : is failure detection rate,

β : is inflection factor.

소개된 Goel-Okumoto, Inflection S-shaped 모델은 고장이 발생 한 후 수정 시 새로운 고장이 발생하지 않는 모델 즉, n_0는 시간에 따라 변화하는 일정한 상수(constant)로 새로운 고장이 소개되지 않는다.

4.3.3 Yamada Imperfect debugging 모델

1992년 Yamada에 의해서 처음 소개된 모델은 고장발생시 고장은 수정되지만 새로운 수정과정을 통해 새로운 고장이 추가되는 고장현상을 모델링하며 식 (3)과 같이 나타낸다.[7)]

$$m(t) = \frac{n_0\lambda}{\alpha+\lambda}(e^{\alpha t}-e^{-\lambda t}), \quad (3)$$
$$n_0(t) = n_0 e^{\alpha t},$$
$$\lambda(t) = \lambda.$$

where

α : is increasing rate of new failure.

4.3.4 Pham-Zhang 모델

1997년 Pham과 Zhang에 의해서 처음 소개되었으며 식 (4)와 같이 나타낸다.[8)]

$$m(t) = \frac{1}{1+\beta e^{-\lambda t}}\left[(c+n_0)(1-e^{-\lambda t}) - \frac{n_0}{\lambda-\alpha}(e^{\alpha t}-e^{-\lambda t})\right], \quad (4)$$
$$n_0(t) = c+n_0(1-e^{\alpha t}),$$
$$\lambda(t) = \frac{\lambda}{1+\beta e^{-\lambda t}}.$$

4.4 매개변수 추정 및 검증

실제 관측된 자료를 이용하여 그래프로 그린 후

이 그래프와 가장 근사한 그래프를 찾을 경우 매개변수는 추정되며 본 논문에서는 최소제곱법을 이용하여 매개변수를 추정하였다.

$$Min\sum_{i=1}^{n}[y_i-\widehat{m}(t_i)]^2. \quad (5)$$

where

y_i : is cumulative failure number at time i,

$\widehat{m}(t_i)$: is cumulative expected failure number at time t.

예컨대, Geol-Okumoto모델에는 n_0, λ 의 매개변수가 있으며 식 (6)에 의해 SSE의 값이 가장 작은 경우 n_0, λ 가 추정되게 된다.

$$SSE(n_0,\lambda)=Min[\frac{D[\sum_{i=1}^{n}[y_i-\widehat{m}(t_i)]^2]}{D\lambda}= \frac{D[\sum_{i=1}^{n}[y_i-\widehat{m}(t_i)]^2]}{Dn_0}=0]. \quad (6)$$

이렇게 추정된 매개변수를 이용하여 가장 적합한 모델을 선택하기 위해서 본 논문은 SSE의 값과 AIC(Akaike Information Criterion) 통계량을 이용하였으며 AIC는 다음과 같이 나타낸다.[8)]

$$AIC=2N-2Log(L) \quad (7)$$

where

N : the number of parameters in the model,

L : likelihood function at its maximum value.

5. Case study

본 논문에서 차량에 내장되어 있는 MOST기반 개인용 전화시스템 2가지를 관찰하였다. 관찰된 대상은 차종만 다를 뿐 모든 기능은 동일하며 MO1, MO2로 명명하겠다. 관찰된 제품의 고장자료를 약 5년간 수집되었으며 MO1의 관찰된 고장 수는 총 4050이며 그 중 소프트웨어에 의한 고장회수는 983이며 MO2의 관찰된 고장 수는 총 2,275이며 그 중 소프트웨어에 의한 고장회수는 792이다.

Table 2, Table 3은 소프트웨어 신뢰성모델을 이용하여 관찰대상 MO1, MO2의 추정된 모수와 SSE 값을 나타내고 있다.

Table 2 Result for MO1

Model name	Parameters	SSE	AIC
G-O model	n_0 = 1056.94 λ = 0.0264237	14503.9	7.5551
Inflection S-shaped	n_0 = 1047.1 λ = 0.0281351 β = 0.0750022	12043.5	9.3684
Yamada	n_0 = 983 λ = 0.0290794 α = 0.00081695	14238.6	10.403
Pham-Zhang	n_0 = 982.997 β = 0.0356445 λ = 0.0289041 c = 63.2496 α = 0.0392944	12021.8	13.3589

Table 3 Result for MO2

Model name	Parameters	SSE	AIC
G-O model	n_0 = 1342.16 λ = 0.0088722	20902.1	13.4654
Inflection S-shaped	n_0 = 1066.27 λ = 0.017177 β = 0.643822	7323.66	14.2203
Yamada	n_0 = 792 λ = 0.0157628 α = 0.00365646	12400.7	16.8105
Pham-Zhang	n_0 = 792.009 β = 0.304755 λ = 0.0183469 c = 297.889 α = 0.0170748	7469.06	18.3483

동일한 기능을 하는 2가지 제품을 관찰한 결과 운영 중 발생한 임베디드 소프트웨어의 경우 SSE의 값이 가장 작은 Pham-Zhang 모델을 잘 따른다고 할 수 있으며 많은 매개변수가 없지만 Inflection S-shaped 모델역시 작은 SSE의 값을 가지는 것으로 알 수 있다. Fig. 3과 Fig. 4는 추정된 모수를 바탕으로 각 신뢰성모델을 묘사한 것이다. Emp는 실제 관측된 누적고장자료를 의미하며 G.O.는 Goel-Okumoto, S는 Inflection S-shaped, Y는 Yamada, P.Z. 는 Pham-Zhang 모델을 의미한다.

하드웨어의 고장의 신뢰성모델링[9)]을 위해 많은 사용되는 분포함수 예컨대, 와이블분포, 대수정규분포, 지수분포 등은 Chi-square test(x^2 test), Kolmo-

gorov – Smirnov test 등과 같은 적합성 검정방법을 통해 적합여부를 확인할 수 있지만 SSE, AIC의 값은 적합여부를 확인할 수 없다는 단점이 있다. 이에 소프트웨어 신뢰성모델의 적합성 테스트를 위한 새로운 방법에 대한 연구가 필요할 것으로 사료된다.

6. 결 론

차량안전과 관련된 법적 규제, 고객의 다양한 요구사항으로 인하여 차량네트워크(CAN, Flexray, Most)들의 연결증가, 소프트웨어 코드라인수의 증가 등 차량의 복잡성은 점점 증가하고 있다. 특히 다양한 기능의 구현을 위해서는 소프트웨어의 사용이 증가하고 있으며 차량 전체 신뢰성향상을 위해서는 이 소프트웨어의 신뢰성향상 가장 중요한 문제로 대두되고 있다. 이에 본 논문에서는 임베디드 소프트웨어 신뢰성평가를 위하여 소개된 신뢰성모델이 운영 중 발생한 고장자료를 이용한 신뢰성평가를 위해 사용가능한가를 살펴보았다. Table 2, Table 3

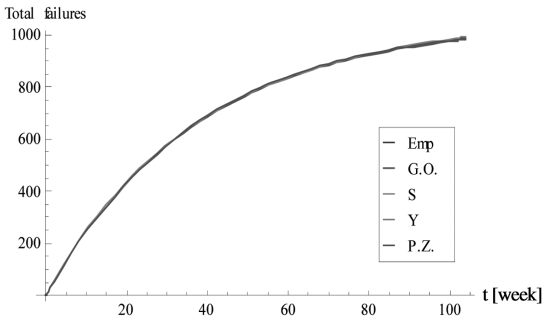

Fig. 3 Empirical failure data and fitted theoretical function for MO1

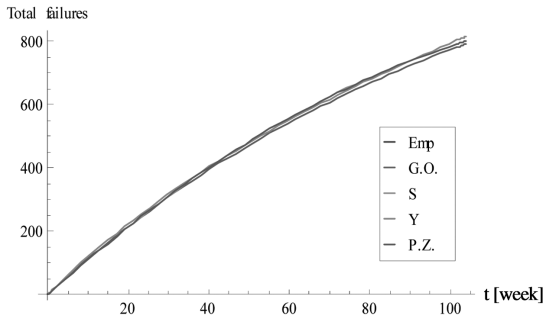

Fig. 4 Empirical failure data and fitted theoretical function for MO2

과 Fig. 3, Fig. 4의 결과 실제 운영 중 발생된 고장자료는 지금까지 소개된 모델들을 잘 따르고 있는 것으로 나타났다. 임베디드 소프트웨어의 경우 실제 운영되는 수량이 많기에 그 만큼 고장발생 확률이 크며 예측할 수 없이 불특정으로 고장이 발생하기에 NHPP기반 소프트웨어 신뢰성모델 역시 운영 중 발생한 고장의 신뢰성특성을 잘 기술할 수 있을 것으로 사료된다.

차량은 크게 파워트레인, 샤시, 바디, 멀티미디어 도메인을 크게 나뉜다. 개발과정에서 각 도메인의 특성에 따라 시스템의 신뢰성 요구사양도 상이하게 된다. 본 논문에서 관찰된 멀티미디어 도메인 시스템의 소프트웨어 신뢰성은 모든 차량 시스템의 신뢰성을 대표할 수 없기에 논문에서 다루지 못했던 타 도메인의 소프트웨어 신뢰성평가 연구도 추후에 필요할 것으로 사료된다.

References

1) Studie HAWK 2015 - Wissensbasierte Veränderung der Automobilen Wertschöpfungskette, McKinsey&Company, 2003.
2) Pressmitteilung 2003, Gesellschaft für Informatik e.V. (GI), 2003.
3) M. A. Hartz, E. Walker and D. Mahar, Introduction to Software Reliability: A State of the Art Review, Reliability Analysis Center, 1996.
4) IEEE Std. 610.12, IEEE Standard Glossary of Software Engineering Terminology, IEEE, New York, 1900.
5) M. S. Kim, S. H. Lee and J. W. Lee, “Reliability Analysis for Train Control System by Software Fault Tolerance Techniques,” Spring Conference Proceedings, KSAE, KSAE09-B02840314, 2009.
6) M. Ohba and S. Yamada, “S-shaped Software Reliability Growth Models,” International Conference on Reliability and Maintainability, 1984.
7) M. Ohba, “Software Reliability Analysis Models,” IBM. J. Research Development, Vol.21, No.4, 1984.
8) H. Pham and X. Zhang, “Comparisons of

Nonhomogeneous Poisson Process Software Reliability Models and Its Applications," International Journal of Systems Science, Vol.31, No.9, pp.1115-1123, 2000.

9) W. G. Shin, S. H. Lee and Y. S. Song, "The Reliability Life Test Design and Analysis of Wiper Motor for Automobiles," Spring Conference Proceedings, KSAE, KSAE06-S0314, 2006.