안전-필수 소프트웨어의 실패모드 정량화에 관한 연구

김영미*, 정충희*, 김현수**
*한국원자력안전기술원
**충남대학교 전기정보통신공학부 컴퓨터전공
e-mail : ymkim@kins.re.kr

A Study on Quantification of Safety-Critical Software Failure Mode

Young-Mi Kim*, Choong-Heui Jeong*, Hyeon Soo Kim**
*Korea Institute of Nuclear Safety
**Dept. of Computer Science & Engineering, Chungnam Nat'l Univ.

요 약

디지털 컴퓨터와 정보처리기술의 급속한 발전과 함께 산업계 전반적으로 아날로그 기술은 쇠퇴하고 디지털 기술로 전환되고 있다. 심지어 안전-필수 기능을 담당하는 원자력발전소의 계측제어시스템에서도 제한적으로 디지털 기술을 채택하여 사용하기 시작했다. 지금까지 소프트웨어의 신뢰도의 정량화에 대한 연구는 많이 이루어져 왔으나 소프트웨어가 가지는 특수성 때문에 연구결과에 대해 전문가들의 동의를 얻지 못하고 있는 상태이다. 원자력발전소에서는 확률적 안전성 평가(PSA)를 수행할 때 소프트웨어의 실패에 기인한 위험은 무시하고 있다. 하지만, 소프트웨어를 기반으로 한 디지털 시스템의 사용이 점점 늘어남에 따라 소프트웨어 신뢰도에 대한 정량화가 점점 더 요구되고 있다. 본 연구에서는 소프트웨어의 실패모드를 정의하고 해당 실패모드에 의해 사고가 발생할 확률을 베이지안 통계이론을 이용하여 정량화하였다.

1. 서론

디지털 컴퓨터와 정보처리기술의 급속한 발전과 함께 산업계 전반적으로 아날로그 기술은 쇠퇴하고 디지털 기술로 전환되고 있다. 심지어 안전-필수 기능을 담당하는 원자력발전소의 계측제어시스템에서도 제한적으로 디지털 기술을 채택하여 사용하기 시작했다. 하지만 소프트웨어를 기반으로 한 디지털 시스템은 아날로그 시스템과는 다른 실패원인 및 실패모드를 가지며, 환경적인 취약점도 지니고 있다. 안전-필수 소프트웨어의 고품질을 보장하기 위해 소프트웨어 생명주기 동안 형상관리, 테스팅 그리고 확인 및 검증 활동과 같은 많은 활동들이 이루어지고 있다. 하지만, 소프트웨어의 신뢰도를 정량화하는 연구결과들은 아직까지 만족스럽지 못한 상황이다[7, 13]. 하지만, 소프트웨어를 기반으로 한 디지털 시스템의 사용이 점점 늘어남에 따라 소프트웨어 신뢰도에 대한 정량화가 점점 더 요구되고 있다.

본 연구에서는 소프트웨어의 실패모드를 정의하고 해당 실패모드에 의해 사고가 발생할 확률을 베이지안 통계이론을 이용하여 정량화하였다.

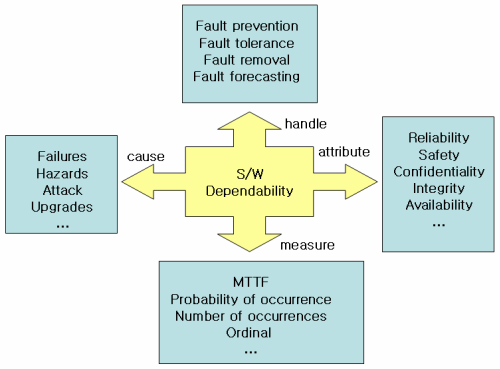
본 논문의 2 장에서는 연구배경을 소개하고, 3 장에서는 일반적인 소프트웨어의 실패모드를 제시한다. 그리고, 4 장에서는 베이지안 이론을 이용한 실패모드 정량화 모델을 제시하고, 5 장에서는 사례연구를 보여준다. 마지막으로, 6 장에서 결론을 맺는다.

2. 연구배경

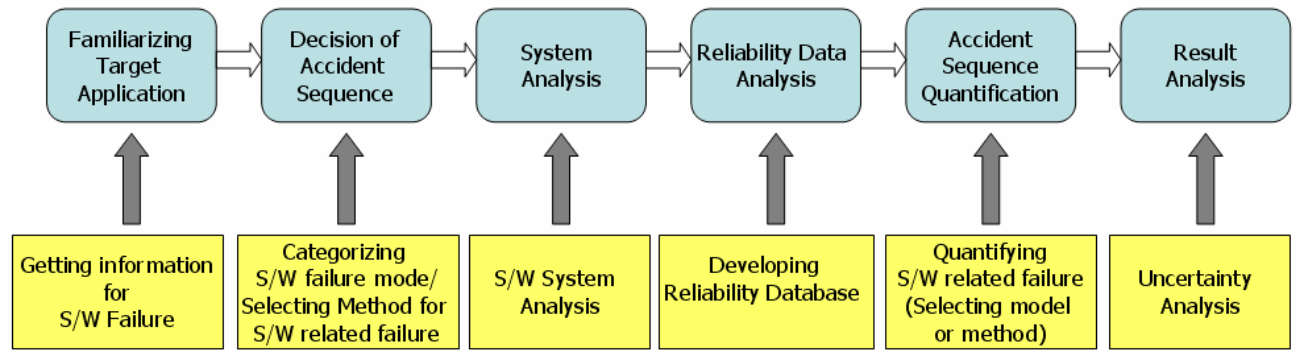
2.1 소프트웨어의 신뢰도(Dependability)

일반 소프트웨어의 신뢰도는 주로 서로 다른 여러 가지 속성들을 포함하는 통합적인 의미로 사용된다. 컴퓨터 시스템의 신뢰도란 시스템이 신뢰할 수 있는 서비스를 제공할 능력을 말한다[1].

(그림 1)은 소프트웨어 신뢰도의 전반적인 특성을 보여준다. 소프트웨어 신뢰도의 속성은 사용자의 요구사항에 따라 다양하게 정의될 수 있으며, 기본적인 속성으로 신뢰성(Reliability), 안전성, 가용성, 무결성 그리고 기밀성 등을 들 수 있다.



(그림 1) 소프트웨어 신뢰도의 전반적인 특성



(그림 2) Level 1 PSA 프로세스와 소프트웨어 관련 태스크

신뢰도는 결함예방, 결함허용, 결함제거 그리고 결함예측 등의 방법을 통해 향상될 수 있으며, 결함 및 실패, 위협(위험을 초래하는 것들), 공격, 업그레이드 등에 의해 손상을 받는다. 또한, 신뢰도의 측정은 대상항목의 특성에 따라 MTTF, 발생확률, 발생 횟수 등으로 가능하다[2][3].

2.2 소프트웨어 실패데이터의 특성

소프트웨어 실패데이터는 아날로그 시스템의 실패데이터와는 매우 다르다. 특히, 안전-필수 소프트웨어의 실패 데이터의 경우는 실패데이터를 얻는 것이 힘든 경우가 많다. 안전-필수 소프트웨어는 실패 발생 건수가 적고 발생한 경우에도 그 결과가 공개되는 경우가 드물다. 또한, 소프트웨어의 실패 데이터는 “객관적”이기도 하고 “주관적”이기도 하다. “주관적”인 실패 데이터의 경우는 전문가의 의견을 통해 특정 데이터 집합과의 연관성의 정도에 대해 의견을 구해야 한다. 베이지안 정량화 접근방식은 이러한 소프트웨어 실패데이터의 특성에 적합하다[4].

3. 소프트웨어 실패모드 모델

3.1 확률적안전성평가(PSA)와 소프트웨어 태스크의 결합

지금까지 원전의 디지털시스템을 위한 PSA 를 수행 시에는 소프트웨어에 의한 위험을 무시하고 수행하였다 [6, 8]. 하지만 소프트웨어 기반의 디지털시스템의 사용이 늘어남에 따라 소프트웨어를 PSA 에 결합시키고자 하는 노력이 시도되고 있다[5]. (그림 2)는 기본적인 Level 1 PSA 프로세스와 소프트웨어 관련 태스크를 결합시켜 보여주고 있다. 소프트웨어를 PSA 에 결합시키기 위해서는 소프트웨어의 실패모드에 대한 분류가 PSA 의 초기단계에서 수행되어야 한다.

3.2 소프트웨어 실패모드 분류

소프트웨어 실패모드는 의도하지 않은 결과로 이르게 하는 초기사건이 무엇인지, 무엇이 의도하지 않는 상태로 가게 하는지에 대한 대답이라 볼 수 있다. 소프트웨어 실패모드는 소프트웨어의 시험 또는 안전성 평가 전에 이루어져야 한다. 특히 소프트웨어 실패모드는 PSA 수행 시에 체크리스트, PHA(Preliminary Hazard Analysis), FMEA(Failure Mode and Effect Analysis), HAZOP(Hazard and Operability Study)를 수행 시에 필요하다.

지금까지 소프트웨어 실패모드에 대한 많은 연구가 이루어져왔다[9,10,11,12,14]. 특히 참고문헌[5]에서는 소프트웨어를 PRA(확률적 위험도 분석)에 적용시키기 위해 소프트웨어 실패모드를 분류하는 연구결과를 제시하였다.

(그림 3)는 디지털 시스템의 일반적인 환경을 보여준다. 소프트웨어는 하드웨어에 둘러싸여 있으며, 또한 주위환경과 인간에 의해 서로 정보를 교환한다. 소프트웨어 실패모드는 상호 중복되지 않고 배타적이 되도록 분류되어야 한다. 본 논문의 소프트웨어 실패모드의 분류는 [5]에서 제시된 실패분류를 참조하여 본 연구의 목적에 맞게 수정한 것이다.

(1) 소프트웨어 기능 실패

소프트웨어 기능 실패 모드는 기능 및 속성의 누락, 잘못된 구현, 불필요한 추가 구현을 포함한다.

(2) 입출력 실패

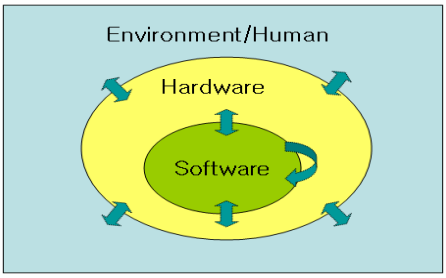
소프트웨어의 입출력 실패 모드는 입출력의 크기, 값, 범위, 형태, 시간, 빈도, 유지기간과 관련이 있다. 부정확한 입출력 변수는 소프트웨어 명세서와 런타임 이미지 사이의 불일치를 초래한다.

(3) 통신 실패

소프트웨어는 다른 소프트웨어, 다양한 하드웨어 및 환경과 상호 통신을 한다. 통신의 실패는 소프트웨어간의 동기화 및 타이밍 실패를 초래한다.

(4) 지원 하드웨어 실패

지원 하드웨어의 실패는 자원 경쟁과 물리적 플랫폼의 실패와 관련이 있다. 자원 경쟁으로 인하여 데드락 및 락아웃이 발생할 수 있으며, 물리적 플랫폼과 관련해서는 CPU, 메모리, 주변장치 등의 실패가 있다.



(그림 3) 일반적인 디지털 시스템 환경

(5) 환경실패

전자기파, 압력, 화재, 기온, 습기 등을 포함하는 환경적인 영향은 소프트웨어가 실행되는 하드웨어 플랫폼에 영향을 줄 수 있다. 하드웨어 플랫폼의 실패는 소프트웨어의 실패를 초래할 수 있다.

4. 소프트웨어 실패모드의 정량화

베이지안 이론은 소프트웨어의 실패 데이터를 특성화하는데 아주 강력한 프레임웍을 제공한다[4]. 본 연구에서는 소프트웨어 실패모드를 베이지안 프레임웍을 이용하여 정량화하였다.

A 를 모든 실패모드의 집합이라고 가정한다.

$A = \{SF_1, SF_2, \cdots, SF_k\}$, k 는 실패모드 분류의 크기이다.

만약 소프트웨어의 실패에 의해 발생한 사건에 대하여 소프트웨어 실패모드 SF_i 에 의해 발생한 사건의 비율을 알고 싶다고 가정을 하자. m 은 $\exists SF_i \in A$ 에 의해 발생한 사건의 수이고, n 은 $\forall SF_j \in A$ 이며, $SF_i \neq SF_j$ 인 사건의 수이다.

$$P(a \mid m,n) = \frac{P(m,n \mid a) \cdot P(a)}{\int P(m,n \mid a) \cdot P(a) da} \quad (1)$$

- a : 가정, 소프트웨어의 실패에 기인한 사건 a 가 발생함.
- $P(a)$: 사전 확률
- $P(a|m,n)$: 조건부 확률, 사후 확률
- $P(m,n|a)$: 조건부 확률, 가능성 확률

이 연구의 목적은 소프트웨어의 실패에 기인하여 발생한 사고 가운데 SF_i 실패모드에 의해 발생한 사건의 비율을 찾는 것이다. 만약 가능성 확률이 이산분포를 따른다면, 수식 (1)은 다음과 같이 고쳐 쓸 수 있다.

$$P(a \mid m,n) = \frac{\binom{n+m}{m} a^m (1-a)^n p(a)}{\int \binom{n+m}{m} a^m (1-a)^n p(a) da} \quad (2)$$

$p(a)$ 에 대한 특정 사전 분포를 이용하게 되면 사후분포를 얻을 수 있으며 적분 값을 구할 수 있다. 예를 들어, $p(a)$가 α 와 β 에 대해 베타 분포를 따른다고 가정을 하면, 사후 분포는 $\alpha + m$ 과 $\beta + n$에 대해 베타 분포를 따르게 된다.

$a \in [0;1]$ 일 때, a의 확률밀도함수는 다음과 같다.

$$P(a) = \frac{a^{\alpha-1}(1-a)^{\beta-1}}{B(\alpha,\beta)} \quad (3)$$

사후 분포는 다음과 같다.

$$P(a \mid m,n) = \frac{a^{\alpha+m-1}(1-a)^{\beta+n-1}}{B(\alpha+m,\beta+n)} \quad (4)$$

수식 (1)-(4)를 이용하면, 특정 소프트웨어의 실패모드에 기인한 사건이 발생될 기대 값을 얻을 수 있다.

5. 사례연구

4 장에서 제시된 정량화 모델을 검증하기 위해 참고문헌[5]에서 Li 가 사용한 데이터를 활용하였다. Li 는 일반인에게 공개되어 있는 항공 시스템의 데이터를 이용하여 소프트웨어 실패 데이터의 분류모델을 검증하였다. Li 는 총 19 개의 사건을 조사하였으며 그 결과는 표 1 에 정리되어 있다.

<표 1> 사고 분류의 예

실패모드	해당 실패모드에 기인하여 발생한 사건의 수
기능 및 속성의 실패	9
입출력 실패	2
통신 관련 실패	0
지원 플랫폼 관련 실패	5
환경에 의한 실패	3

[5]에서는 각 사고들의 원인을 파악하여 소프트웨어의 실패모드에 적용하였다. 그 결과 기능 및 속성의 실패로 인한 사고가 9 건, 입출력 실패에 기인한 것이 2 건, 통신과 관련된 것이 0, 메모리 및 CPU 등 지원 플랫폼 실패에 의한 것이 5 건 그리고 환경적 요인에 의한 것이 3 건이었다. 전체 사고의 절반이 소프트웨어의 기능적인 실패에 의해 발생하였으며, 나머지 절반이 물리적 및 환경적인 요인에 의해 발생하였다. 표 1 은 실패모드의 형태와 해당하는 실패모드에 의해 발생한 사건의 건수를 보여준다.

위의 예제에서 소프트웨어의 실패에 기인하여 사건이 발생한 경우를 a 라고 가정할 수 있다. 우리가 소프트웨어의 기능 및 속성의 실패에 의해 사건이 발생한 경우에 대하여 관심이 있는 경우 m=9, n=10 의 값을 구할 수 있다. 이 값들을 수식 (1)에 적용을 시키면 아래와 같다.

$$P(a \mid 9,10) = \frac{P(9,10 \mid a) \cdot P(a)}{\int P(9,10 \mid a) \cdot P(a) da}$$

$$= \frac{\binom{19}{9} a^9 (1-a)^{10} p(a)}{\int \binom{19}{9} a^9 (1-a)^{10} p(a) da}$$

$$= 1847560 * a^9 (1-a)^{10} \qquad (5)$$

수식(5)는 a 에 대한 사후확률함수이다. 만약 우리가 소프트웨어의 실패에 기인하여 발생한 사건들 가운데 소프트웨어의 기능 및 속성의 실패에 의해 발생한 사건이 절반 이상인 경우에 관심이 있다면, 우리는 사전 확률을 1/2 로 하여 계산할 수 있다.

$$1847560 \int_{1/2}^{1} a^9 (1-a)^{10} da \approx 0.41$$

이 경우 소프트웨어의 기능 및 속성의 실패에 의해 사건이 발생할 확률이 대략 41%임을 예측할 수 있다.

6. 결론

본 논문에서는 일반적인 소프트웨어 실패모드 분류기준과 베이지안 이론을 이용하여 소프트웨어 실패모드를 정량화하는 모델을 제시하였다. 안전-필수 소프트웨어의 실패 데이터의 확보가 힘들어 공개된 항공시스템의 실패데이터를 이용하여 모델의 사례연구를 수행하였다. 제시된 본 모델은 원전과 같은 안전-필수 시스템에 소프트웨어를 포함하여 PSA 를 수행할 때 적용될 수 있을 것으로 기대된다. 또한, 보다 많은 사례연구를 통해 소프트웨어의 시험단계에 어느 실패모드에 대한 시험에 보다 더 많은 노력을 기울어야 하는지에 대한 정보도 얻을 수 있으리라 기대된다.

참고문헌

[1] Laprie, J.-C., *Dependability: basic concepts and terminology*, dependable computing and fault-tolerant systems. Springer Verlag, Wien-New York.

[2] Paolo Donzelli, Victor Basili, A practical framework for elicting and modeling system dependability requirements: Experience from the NASA high dependability computing project, The Journal of Systems and Software, 2005

[3] A framework for dependability engineering of critical computing systems, Mohamed Kaaniche, Jean-Claude Laprie, Safety Science, Volume 40, December 2002

[4] C. Smidts, D. Sova, *An architectural model for software reliability quantification: sources of data*, Reliability Engineering and System Safety, 1999.

[5] Bin Li, Ming Li, Susmita Ghose, Carol Smidts, *Integrating Software into PRA*, International Symposium on Software Reliability Engineering, 2003

[6] S. A. Arndt, N. O. Siu, &E. A. Thornsbury, *What PRA Needs From A Digital Systems Analysis*, 2002

[7] Norman E. Fenton, *A Critique of Software Defect Prediction Models*, IEEE Transaction on software engineering, Vol. 25

[8] C. G. Park, J. J. Ha, *Probabilistic Safety Assessment*, 2003, www.brainbook.net

[9] A. D. Gordon, *Classification*, 2nd edition, Chapman & Hall, 1999

[10] C. Smidts, M. Stutzke, R. W. Stoddard, *Software Reliability Modeling: An Approach to Early Reliability Prediction*, IEEE Transactions on Reliability, Vol. 47, No. 3, 1998, Sept. pp268-278.

[11] R. R. Lute, *Analyzing Software Requirements Errors in Safety-Critical Embedded Systems*, Proceedings of IEEE International Symposium on Requirement Engineering, 1992

[12] P. L. Goddard, *Software Reliability Technique*, Annual Reliability and Maintainability Symposium, 2000

[13] T. Y. Sung, H. S. Eom, *A study on the quantitative evaluation for the software included in digital systems on nuclear power plants*, 2002, KAERI/TR-2091

[14] Software Preliminary Hazard Analysis for the UNCHIN Nuclear Power Plant, Unit 5 and 6, rev0, 2001