2000 년 추계학술발표 논문집
한국원자력학회

PSA를 이용한 원전 안전계통 소프트웨어 시험횟수 결정

Determination of the Number of Software Tests Using Probabilistic Safety Assessment

강현국, 성태용, 이기영

한국원자력연구소
대전광역시 유성구 덕진동 150

요 약

디지털 기기의 원전에의 적용이 일반화되면서 소프트웨어의 안전성에 관한 문제가 제기되고 있는데, 소프트웨어의 고장은 적용 환경과 대상에 대해 비선형적인 특징을 나타내는 것으로 알려져 있어, 원전의 안전계통에의 실제 적용을 위해서는 사전시험이 필수적이다. 안전관련 시스템에서의 소프트웨어 사전시험 결과는 일반적으로 오류를 포함하고 있지 않으므로 몇회의 시험을 오류발견 없이 수행해야 소프트웨어가 만족할 만한 신뢰도를 가진다고 할 수 있는지를 결정하기가 어렵다. 본 연구에서는 전체 시스템의 확률론적 안전평가 결과로부터 소프트웨어가 만족해야 할 불가용도 수준을 추정하고, 이를 이용하여 사전시험 횟수를 결정하는 방법을 제안하였다.

Abstract

The broader usage of digital equipment in nuclear power plants gives rise to the safety problems of software. The field test should be performed before the software is used in critical applications because it is well known that software shows non-linear response when it is applied to different target systems in different environment. In the case of safety-critical applications, the result of tests contains usually zero failure case and the satisfiable number of tests is hard to be determined. In this paper, we suggests the method to determine the number of software tests without failure using the probabilistic safety assessment. From the result of the probabilistic safety assessment on total system, the desirable unavailability of software is calculated and the number of tests is determined.

1. 서론

디지털 계측제어기기의 사용이 일반화된 일반 산업계와 달리 원자력 발전소에서는 극히 제한적으로 사용되어 왔으나, 기존의 아날로그 기기는 부품 조달이 원활하지 못해 더 이상의 유지·보수에 어려움이 많고 디지털 기기를 이용할 경우 단순화·표준화로 인한 설계와 유지보수의 편의성 향상 등 많은 장점이 있으므로 신규 원전에 디지털 기기를 도입하는 폭이 점차 넓어지고 있다. 국내에서도 신규 건설중인 울진 5, 6호기 및 차세대원전의 원자로 보호계통과 공학적안전설비 작동계통과 같은 안전기능을 수행하는 계통에 디지털계측기술을 적용하고 있으며, 가동중인 원전에 대해서도 노후 기기에 대한 디지털 기기로의 대체가 추진되고 있다. 이에 따라 디지털 기기의 안전성에 대한 관심이 높아졌으며, 특히 소프트웨어의 안전성 입증에 관한 문제가 제기되고 있다. 최근에는 해외 원자력계에서도 소프트웨어의 오류가 전체 시스템 고장의 중요한 요인이 된다는 연구 결과들이 보고되고 있는데, 미국의 운전 경험에 의하면 디지털 계측제어 시스템의 전체 고장 중 38% 정도가 소프트웨어의 문제에 기인하는 것으로 밝혀졌다고 하며, 캐나다의 경우 35% 정도가 소프트웨어의 문제에 기인하는 것으로 밝혀졌다[1].

하드웨어의 고장율을 정량화하는 방법론이 이미 잘 개발되어 활용되고 있는 것과는 달리, 소프트웨어 오류에 의해 유발되는 고장율을 추정하는 방법에 관해서는 아직도 많은 논란이 있다. 소프트웨어에 의한 시스템의 고장 유발에 관련한 연구는 상대적으로 그 역사가 짧기 때문에 충분한 자료가 축적되어 있지 못하다. 따라서 정량적인 분석의 기초가 된다고 할 수 있는 고장시의 거동에 대한 분석이 충분히 수행되지 못하여 여러 가지 논란을 일으키고 있다. 소프트웨어의 특성상 통계적으로 다룰 수 있는 문제인지 자체가 논란의 대상이 되고 있어, 일반적으로 공감하는 평가방법론이 완전히 정착되기까지는 많은 시간이 걸릴 것으로 판단된다.

그러나 이러한 논란과는 별도로, 원자력발전소의 안전계통과 같이 안전성이 극히 중요시되는 분야에의 실제 적용을 위해서는 소프트웨어의 사전시험이 필수적인 것으로 판단된다. 따라서 본 연구에서는 소프트웨어의 사전시험의 결과에 중점을 두어 소프트웨어의 고장을 확률적인 문제로 다루었다. 물론 확률적인 시험만으로 소프트웨어의 신뢰도를 완벽히 추정할 수 있지 않다는 것이 소프트웨어공학계의 일반적인 입장이며, 따라서 생명주기에 걸친 철저한 형상관리 및 확인검증의 여부 등 소프트웨어의 정성적 분석이 신뢰도에 반영되어야 할 것이나, 이와 관련하여 현재 정립된 방법론이 없으므로 정성적 분석은 논외로 하였다.

안전관련 시스템에 이용되는 소프트웨어의 사전시험 결과는 일반적으로 오류를 포함하고 있지 않다. 다시 말하자면, 오류가 발견된 소프트웨어를 안전관련 시스템에 그대로 적용하지는 않는다는 것이다. 그러므로 몇회의 시험을 수행해야 소프트웨어가 만족할 만한 신뢰도를 가진다고 할 수 있는지가 불분명하다. 많은 시험을 수행할 수록 높은 신뢰도를 가지겠지만, 경제성과 시간을 고려할 경우 현실적인 방법이 아니며, 적절한 시험횟수를 결정하기 위한 보다 합리적인 근거가 필요하게 된다. 본 연구에서는 전체 시스템의 확률론적 안전평가 결과로부터 소프트웨어가 만족해야 할 불가용도 수준을 추정하는 방법을 제안하였으며, 이를 예제 시스템에 적용하여 사전시험 횟수를 결정하는 것을 보였다.

2. 소프트웨어의 고장 확률

전술한 바와 같이, 최근 발간된 해외의 보고서에서는 디지털 시스템의 오류중 3 분의 1 정도가 소프트웨어에 기인하는 것으로 밝히고 있다. 즉, 하드웨어만으로 전체 시스템의 불가용도를 추정할 경우 상당히 부정확한 결과를 얻게 되는 것이다. 이에 따라 소프트웨어의 안전성에 대한 관심이 고조되고 있지만, 현재로는 소프트웨어에 대해서는 엄격한 명세서와 그 이행사항을 반영한 고품질의 개발 공정을 채택했는지의 여부를 중심으로 정성적인 평가만이 가능하며, 최종 제품의 필수 기능을 검증하는 정도의 실제 사전시험만이 이루어지는 것이 현실이다.

소프트웨어 고장의 발생 여부, 소프트웨어 고장 발생의 무작위성 여부, 소프트웨어 고장률의 개념 유무는 현재 소프트웨어공학계 내부에서도 논란이 되고 있는 사안이다. Laplace 등의 연구자들은 소프트웨어의 경우 개발 과정에 대한 철저한 관리만으로도 충분한 수준의 신뢰도를 확보할 수 있다고 주장[2]하고 있는데, 실제 원전 운전 경험을 토대로 했다는 점에서 신빙성이 있으나, 최근의 디지털 계측제어 기기들에 이용되는 소프트웨어의 복잡도가 예전에 비해 크게 높아졌다는 점과 디지털 기기들이 안전성에 미치는 영향이 증대되고 있다는 점을 고려할 때, 실제 사전시험 없이 소프트웨어가 충분한 신뢰성을 갖추었다고 주장하는데는 무리가 있을 것으로 판단된다.

일부 학자들은 소프트웨어의 수행이 틀린 결과를 산출하더라도 물리적 변화가 발생하는 것은 아니므로 소프트웨어 고장은 일어나지 않는다고 주장하며, 다른 한편에서는 소프트웨어는 성공적으로 실행되거나 그렇지 않거나 둘 중의 하나이므로, 그 신뢰도는 0 이거나 1 이 된다고 주장하기도 한다 [3]. 소프트웨어 고장 개념을 수용할 지라도 입력과 내부상태가 주어진 상황에서 소프트웨어의 동작은 고정되어있기 때문에 소프트웨어는 결정론적이라고 논하기도 한다.

소프트웨어는 그 고장의 본질상 결정론적인 요소를 많이 가지고 있다. 즉, 소프트웨어의 명세오류나 코딩 오류는 처음부터 그 소프트웨어에 내재되어 있던 것이므로 고장은 결정론적인 양상을 보이게 되는 것이다. 한편, 이러한 점을 인정하면서도 소프트웨어를 확률론적으로 다룰 수 있다는 주장도 있다. 즉, 실제 적용을 위해서 소프트웨어 오류 발생의 무작위성을 인정하는 '고장률'을 이용하는 경우가 많은 것이다. 이것은 소프트웨어에 대한 고장률 개념은 '빈도'가 아니라 '기대치(확신도)'의 개념이기 때문이다. 테스트 결과 한번도 오류를 일으킨 적이 없는 소프트웨어가 잠재되어 있던 문제점으로 인해 '앞으로의 실제 사용시 고장을 일으킬 것으로 기대하는 정도'를 추정하는 문제가 되기 때문이다. 이러한 무작위성 가정에 대해서 'error crystal'의 개념을 이용하여 정당성을 설명하는 것이 일반적이다. 즉, 소프트웨어의 잠재된 오류가 특정부분(error crystal)에 숨어 있다가 입력값이 그 부분을 활성화시킬 경우에 비로소 외부로 오류가 드러난다는 것인데, 이 입력값의 무작위성으로 인해 소프트웨어가 오류를 발생시키는 양상이 무작위성을 띄게 된다는 것이다 [3].

일단 소프트웨어의 신뢰도를 무작위성을 가지는 고장률의 개념으로 취급할 수 있다고 가정하면, 고장률의 추정이 필요하게 된다. 하드웨어의 경우에는 각 부품들의 고장률에 관한 많은 표준[4]을 참고할 수 있으나 소프트웨어는 이 같은 것이 없으므로 직접적인 사전시험이 필요하게 된다. 원자력발전소의 안전계통에 오류가 발견된 소프트웨어를 그대로 적용하는 것을 상정하는 것은 비현실적이다. 따라서 시험을 몇회 수행하던지 간에, 소프트웨어 관련하여 발생한 오류가 없는 결과를 얻게 될 것이다. 위에 언급한 바와 같이 '앞으로의 실제 사용시 고장을 일으킬 것으로 기대하는 정도'를 추정하기 위한 기본 자료로 사전시험 횟수를 사용하는 것이다.

이러한 정량적 분석방법의 적용이 타당한지는 수행된 사전시험의 속성에 달려있다. 사전시험의 입력은 실제로 직면하는 입력들을 대표해야 하고, 모든 경계 조건들과 잠재된 모든 위험들을 확실히 내포하고 있어야 하며 각 시험 입력들은 서로 독립적이어야 한다. 그러나 실제로 이러한 조건을 완벽히 만족하기는 어려우므로, 사전시험은 안전성 평가와 품질보증을 위해 적용되는 전체 과정의 한 부분으로 사용되어야 할 것으로 판단된다. 정량적 분석과 정성적 분석은 서로 상호 보완적인 관계에 있다.

소프트웨어의 신뢰도에 대한 정성적 분석은 소프트웨어의 결정론적 측면을 강조한 것이다. 따라서 생명주기에 걸쳐 철저한 형상관리가 이루어졌는지, 확인검증이 충분한지 등을 고려한다. 한편 정량적 분석은 소프트웨어 오류를 확률론적인 측면에서 분석하는 것이며 이는 실제 가동 시험의 결과에 대한 분석을 통해 이루어진다. 본 연구에서는 소프트웨어의 사전시험에 중점을 두어, 소프트웨어의 고장을 확률론적인 문제로 다루었다. 즉, 소프트웨어 고장의 무작위성을 인정하는 것이다. 그러므로 이러한 접근에는 여러가지 전제가 필요하며, 또한 그 결과물의 용도도 제한적이다. 확률적인 시험만으로 소프트웨어의 신뢰도를 완벽히 추정할 수 있지 않다는 것이 소프트웨어 공학계의 일반적인 입장이지만, 소프트웨어의 정성적 분석의 결과를 신뢰도에 반영하는 것과 관련하여 현재 정립된 방법론이 없으며 연구가 진행중 이므로 정성적 분석은 논외로 하였다 [5].

3. 디지털 안전계통의 확률론적 안전평가 (PSA)

원전의 안전성을 종합적이며 정량적으로 평가하기 위한 중요한 안전성평가 수단으로 확률론적 안전평가(probabilistic safety assessment; PSA)가 사용되고 있으며, 신규 원자력 발전소 건설시 인허가 사항으로 제출이 요구된다. PSA 는 논리적으로 이상사건에 대한 발전소 대응을 모델하며, 이를 통해 각 사고경위의 원인 및 발생빈도를 기기의 단위까지 파악 할 수 있으며, 각 계통의 주어진 기능을 수행실패에 대해서도 원인과 각 확률 값을 구할 수 있다. 이러한 결과는 원하지 않는 사건의 발생 확률과 원인을 밝힐 수 있기 때문에 이를 이용하여 설계 검증, 정비 최적화 등에 다양하게 이용된다.

PSA 는 인허가 측면에서뿐만 아니라 원전의 설계 및 운전을 포함하여 원전의 안전성을 종합적으로 평가하는 도구로써 1980 년대부터 사용이 확대되어왔다. 원자력 발전소에서 일어날 수 있는 가상사고에 대해 발전소 설비 및 운전원의 대응을 논리적인 모델로 구축하여 노심손상을 유발하는 사고시나리오를 도출하고 그 발생 가능성 및 결과를 평가하여 대상 원전의 안전성을 평가한다. 발전소 모델 구축 시에는 초기사건을 완화시킬 수 있는 비상노심냉각계통과 같은 안전 기능들과 이들의 운전에 필요한 계측제어나 전력 계통과 같은 지원 계통도 모델되어 각 계통의 신뢰도를 구할 수 있다. 최근에는 미국을 중심으로 PSA 결과를 결정론적인 규제의 보완 수단으로 사용하고 있으며, 국내에서도 이의 채택이 적극적으로 추진되고 있다. 또한 PSA 는 초기 설계 단계에 적용되어 설계 검증에 이용되며 설계 개선에도 활용될 수 있다.

일반적인 원전에 대한 모델 구축은 초기사건 발생 이후 이를 완화시키는 계통들의 대응을 모델한 사건수목(event tree)과 각 계통의 요구되는 기능 수행을 나타내는 고장수목(fault tree)으로 구성된다. 이들 모델을 정량화하여 사고경위나 계통의 기능 실패에 대해 발생원인을 나타내는 최소단절집합(minimal cutset)과 이들의 발생 빈도나 확률을 구한다. 계통 설계 시에는 이들 결과로부터 기

능 고장을 유발하는 고장 원인과 그들의 순위를 파악함으로써 설계의 타당성을 입증할 수 있으며, 설계의 취약점을 파악하여 개선안을 도출할 수 있다. PSA 에서 의미 있는 결과를 도출하기 위해서는 발전소의 사고에 대한 대응이나 계통의 기능과 물리적 구성 상태를 정확히 모델링하여야 하며, 사용되는 신뢰도 자료가 정확하여야한다.

원자력 발전소의 경우와 같이 안전성이 중요시되는 시스템에서는 안전 관련 계통을 다중으로 처리하게 된다. 안전 관련 디지털 계측제어 시스템의 경우에는 전체 시스템에서 다중성을 가질 뿐만 아니라 내부의 소프트웨어나 하드웨어에서도 다중성을 가지거나 자기 감시 등 여러 가지 고장내구성 기법(fault- tolerant feature)들이 적용된다. 이러한 기법들이 적절히 적용될 경우, fail-safe 개념에서 큰 신뢰도 향상을 기대할 수 있다. 전체 시스템의 신뢰도 분석에 있어서 이러한 다중·자기감시 기능의 신뢰도는 대단히 큰 영향을 미친다. 일반적으로 디지털 계측제어 시스템에 고장내구성 기능을 추가함으로 인해 시스템의 복잡도가 높아지더라도, 그것은 자기검사로 인한 신뢰도 향상의 장점을 상쇄할 만한 것은 아니기 때문에 이러한 자기감시 기법의 적극적인 도입이 강조되고 있다. 물론, 이러한 기법의 적용이 안전성 향상에 기여함을 보이기 위해서는 정확한 평가 방법론의 개발도 병형되어야 한다.

기존 원전에 적용되어 왔던 PSA 방법론을 그대로 최근의 디지털 기기에 적용하여 안전성을 정량화하는 데에는 몇가지 문제점이 있다 [6]. 디지털 기술의 특성상 기존의 PSA 에서 모델이 힘든 부분이 있으며, 대표적인 것으로는 소프트웨어에 대한 신뢰도 평가, 동적인 디지털 기술 특성 모델 방법, 고장내구성 기법의 평가 등이 있다. 이외에도 정량평가 방법론과 관련된 고장 유형의 평가, 신뢰도 자료의 타당성, 초기사건 모델, 공통원인 고장(common cause failure; CCF) 모델 방법 등과 같은 다양한 문제를 해결하여야 한다.

소프트웨어의 경우, 코드의 공유나 데이터의 공유가 활발해짐으로써 공통원인 고장의 우려를 높이고 있다. 안전성 향상 설계로 전술되었던 다중화의 경우에도, 동일 소프트웨어를 사용할 경우에는 공통원인 고장의 원인이 된다. 이러한 공통원인 고장의 영향은 원자력 발전소의 안전을 위해 필수적인 다중성의 확보를 저해하게 되므로 이를 방지하기 위해 노력이 요구된다. 소프트웨어를 작성할 때 단일 알고리듬만을 적용하는 것이 아니라 다중의 알고리듬을 가지도록 작성할 수 있으며, 다중화된 하드웨어에 각각 다른 소프트웨어를 적용할 수도 있다. 이러한 노력들은 응용 소프트웨어에의 고장내구성 기법 적용 사례에 해당하는데, 이와 같은 기능들은 신뢰성에 매우 큰 영향을 미치기 때문에 이들을 정량평가에 적극적으로 반영하여야 한다.

4. 소프트웨어 사전시험 횟수 결정방법 및 적용예제

대상 소프트웨어의 불가용도가 전체 계통의 불가용도에 어떤 영향을 미칠지를 파악할 수가 없으므로 소프트웨어의 목표 불가용도를 직접 설정하는 것은 어렵고 복잡한 작업이다. 따라서 전체 계통에 어떠한 영향을 미치는 지를 고려하기 위해 PSA 를 이용하였다. 전술한 바와 같이 PSA 는 인허가 측면의 원전의 안전성을 종합적으로 평가하는 도구일 뿐만 아니라 초기 설계 단계에 적용되어 설계 검증에 이용되며 설계 개선에 활용되기도 한다. 즉, 신뢰도 배분의 문제를 해결하는 유용한 의사결정 도구로 이용될 수 있는 것이다. 본 논문에서는 PSA 의 이러한 측면을 이용하여, 안전관련 소프트웨어의 사전시험 횟수를 결정하는 방법론을 제시하고자 한다.

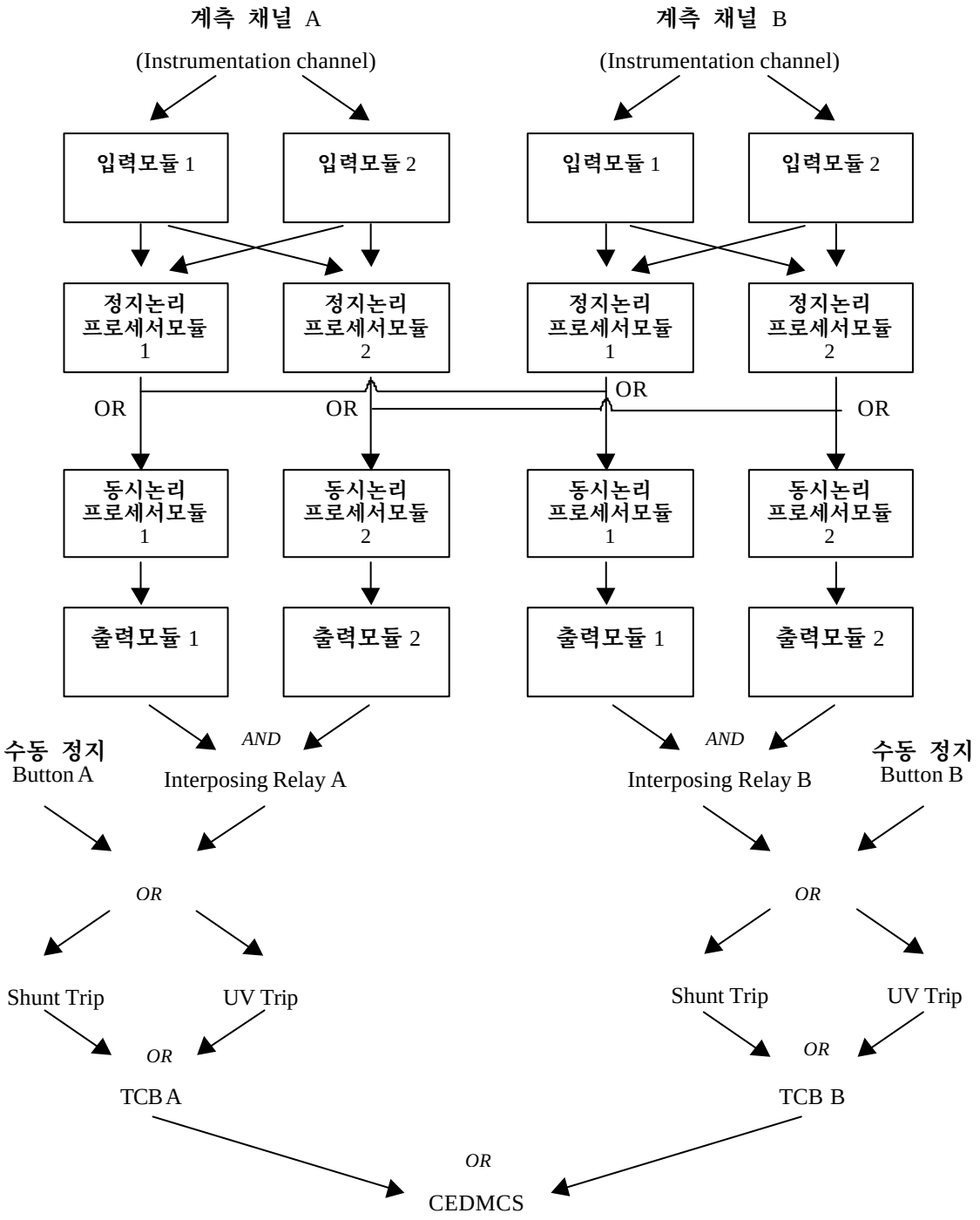
소프트웨어의 사전시험 횟수를 결정하기 위한 작업의 순서는 다음과 같다. 원자력계에서는 계통의 신뢰도를 구하기 위하여 고장수목 기법이 많이 사용되므로, 본 논문에서도 고장수목을 이용한 방법을 제안하였다. 그러나 반드시 고장수목을 이용할 필요는 없으며 다른 분석 기법을 이용하는 것도 가능하다.

- 적용대상을 포함한 분석의 대상범위를 설정한다.
- 소프트웨어를 포함한 대상 계통의 고장수목 모델을 작성한다.
- 작성된 고장수목 모델을 이용하여 주요 최소 단절집합을 추출한다.
- 계통의 불가용도 목표치를 설정한다.
- 추출된 단절집합 확률의 합이 목표 불가용도를 넘지 않도록 소프트웨어 모듈의 목표 불가용도를 계산하고 그에 따라 사전시험 횟수를 결정한다.

본 논문에서는 <그림 1>에 도시한 바와 같은 가상의 원자로 보호계통의 정지 논리 및 동시 논리 부분을 디지털화하여 소프트웨어로 적용하는 예제를 들었다. 예제의 보호계통은 <그림 1>에 도시한 바와 같이, 2개의 정지 채널로 한정하였다는 점을 제외하고는 실제 원자력 발전소에 적용되는 보호계통의 구성과 거의 유사하다. 정지 채널의 갯수, 채널내 다중성의 확보, 입출력 모듈의 배치, 프로세서모듈의 갯수 등은 설계자가 목적에 맞게 변경할 수 있는 것이므로, 본 논문에서는 다음과 같은 특징을 가지는 예제 계통만을 설명하기로 한다.

두개의 정지 채널은 현장입력 신호부터 Trip circuit breaker (TCB)까지 정지논리 프로세서와 동시논리 프로세서간의 통신을 제외하고는 완전히 독립되어 있다. 그림에서 사각형 박스내에 표시된 부분들은 정지 및 동시 논리 계통에 속하는 모듈들을 나타낸다. 각 채널은 운전원의 수동정지를 위한 버튼을 포함한다. 두개의 TCB 가 모두 차단되어야 전원이 차단되어 제어봉이 삽입되도록 되어 있다. 또한, 각 채널의 내부에 있는 모든 모듈은 2 중화되어 있다. 다만 입력모듈의 출력값은 두개의 프로세서 모듈에 의해 공유된다. 채널 내의 다중성에 대해서는 두가지의 경우를 생각할 수 있는데, 첫째는 다중성을 유지하기 위하여 동일한 역할을 하는 모듈은 서로 다른 제조업체에서 별개의 공정을 거쳐 생산된 것을 사용하여 두 모듈간의 공통원인 고장을 무시할 수 있도록 구성한 경우이고, 둘째는 동일 기능 모듈을 같은 제조사의 제품을 사용하여 공통원인 고장 가능성이 있도록 구성하는 경우이다. 본 예제의 분석에서는 첫번째의 경우만을 분석하였다.

계측 채널의 불가용도는 각 계측기의 고장 또는 공통원인 고장 그리고 계측 채널 우회의 3 가지 경우로 한정한다. 그리고 일반적으로 디지털 시스템에서 마이크로프로세서와 소프트웨어를 포함하는 프로세서 모듈에는 감시 타이머(watchdog timer)를 부착하여 오동작을 감시하는 경우가 많다. 원자력 발전소와 같이 안전성이 중요시되는 분야에서는 특히 이런 고장내구성 기법의 적용이 필수적이다. 따라서 그림에는 표시하지 않았지만, 본 예제에서도 감시 타이머가 동시논리 프로세서 모듈을 독립적으로 감시하여 heart-beat 을 점검하고, 만약 정상적인 heart-beat 이 감지되지 않을 경우에는 Interposing relay 로 연결되는 출력을 차단하여 정지 신호를 발생하도록 설계된 것으로 가정하였다. 한편 정지논리 프로세서는 동시논리 프로세서가 통신을 통해 감시하는 것으로 가정하였다.



<그림 1> 예제의 디지털 원자로 보호계통 구성

(두개의 채널을 가지고 있으며 각 채널 내부에는 2 개씩의 다중 모듈 가지며 사각형 박스는 정지 및 동시 논리 계통에 속하는 모듈을 표시한다)

(1) 대상 계통 설정

먼저 대상 단위 계통을 설정하여야 한다. 현장의 계측 채널부터 TCB 까지의 범위를 분석의 대상으로 설정할 수도 있고, 정지 논리 및 동시 논리 계통에 속하는 부분만을 분석의 대상으로 설정할 수도 있다. 설정한 범위의 계통에 대해 분석을 수행하고 목표 불가용도를 결정하여야 하는데, 본 예제에서는 현장의 계측채널부터 TCB 까지의 범위를 단위 계통으로 설정하였다.

(2) 대상 계통의 고장수목 모델 작성

고장수목을 작성하는 것은 특정 정지요구 상황 발생시에 제어봉 구동장치의 전원을 차단하지 못할 확률을 정량적으로 평가하기 위한 것이다. 안전성 분석을 위한 모델이 실제의 복잡한 모습을 그대로 정확히 반영할 수는 없으므로, 몇가지의 가정을 통해 단순화를 하게 된다. 모델링의 규모가 적절하게 한정되어야 하고 정적(static)인 모델을 이용하여 동적(dynamic)인 실제 시스템을 표현하여야 하기 때문에 가정의 도입 없이는 현실적인 분석이 불가능하다. 이러한 고장수목 모델링의 주요 가정을 나열하면 다음과 같다.

- 증기발생기 저수위 상황에 한정하여 증기발생기 수위, 가압기 압력의 두가지 계측 변수만을 고려하여 분석을 수행하였다. 이 두 계측변수는 입력모듈 1 과 2 에 분산되어 연결된 것으로 가정하였다.
- 프로세서간의 통신 및 프로세서와 입출력 모듈간의 통신 과정에서의 계통의 기능상실 고장이 발생할 가능성은 무시하였다.
- 동일 기종 기기간의 공통원인 고장만을 고려하였으며, 이기종 프로세서 및 입출력 기기간의 공통원인고장은 무시하였다. 동일 기종 프로세서간에는 소프트웨어 개발 도구 등이 동일하므로 소프트웨어 오류가 공통원인 고장으로 작용하는 것으로 가정하였다.
- 각 기기는 1 개월의 점검 주기를 가지는 것으로 가정하였다.
- 감시타이머가 동시논리 프로세서에서 발생하는 하드웨어적 오류를 모두 검출할 수 있고, 동시논리 프로세서가 비교논리 프로세서에서 발생하는 하드웨어적 오류를 모두 검출할 수 있는 것으로 가정하였다. 소프트웨어 오류는 감시타이머가 검출할 수 없는 것으로 보수적으로 가정하였다.
- 동시에 1 개의 계측채널만 우회가 가능한 것으로 가정하였다.

이러한 가정을 도입하여 <그림 1>의 구성에 대해 작성된 고장수목 모델은 <그림 2>에서 <그림 5>까지에 도시하였다. 고장수목 작성을 위하여 한국원자력연구소에서 개발한 신뢰도 분석 지원 프로그램인 KwTree 를 이용하였다.

이러한 기능적, 구조적 모델링뿐만 아니라 기본사건의 확률을 구하는 것도 중요한데, 본 예제에서는 TCB, 센서 등 기존의 아날로그 장치에 대해서는 종래의 분석과 같은 값을 적용하고, 디지털 기기에 대해서는 <표 1>의 값을 임의로 할당하였다.

(3) 최소단절집합 추출

위에서 작성된 고장수목에 대하여 모든 최소단절집합을 구하는 것은 비현실적인 일이다. 따라서 주요 최소단절집합만을 구하여 분석한다. 소프트웨어의 고장이 주요 최소단절집합에 포함될 수 있도록 우선은 '소프트웨어 고장'의 기본사건에 상대적으로 높은 고장확률을 설정하여 최소단절집합을 생성한다. 예제에서는 한국원자력연구소에서 개발한 최소단절집합 생성 프로그램인 Kcut 을

<표 1> 디지털 기기의 고장율

기기명	시간당 고장율	고장 확률
Digital Output Module	2×10^{-6}	7.22×10^{-4}
CCF of Digital Output Modules	2×10^{-8}	7.22×10^{-6}
Processor Module	4×10^{-6}	1.44×10^{-3}
CCF of Processor Modules	1×10^{-7}	3.61×10^{-5}
Analog Input Module	3×10^{-6}	1.08×10^{-4}
CCF of Analog Input Modules	2×10^{-7}	7.22×10^{-5}
Watchdog Timer Contact	-	1.00×10^{-7}
CCF of Watchdog Timer Contacts	-	1.00×10^{-8}

이용하여, 소프트웨어의 고장율을 1×10^{-5}으로 설정하고 최소단절집합을 생성하였다. 10^{-12}이상의 확률을 가지는 최소단절집합만을 구하였다. 생성된 주요 최소단절집합 중 10^{-9}이상의 확률을 가지는 것들만 <표 2>에 보였다.

(4) 대상 계통의 불가용도 목표 설정

소프트웨어를 제외한 대상 계통의 불가용도는 위에서 구한 최소단절집합의 확률의 합에서 소프트웨어와 관련된 부분을 제외하면 쉽게 구할 수 있다. 또는 최소단절집합 생성시에 소프트웨어의 고장율을 0으로 하면 주요 최소단절집합에는 소프트웨어의 고장은 포함되지 않게 된다. 예제 계통의 경우에는 3.42×10^{-6}으로 나타났다. 분석의 대상 계통이 TCB, 센서 등을 모두 포함한 것임을 고려할 때, 소프트웨어로 인한 불가용도의 증가가 최소화되어야 할 것이다. 원전 전체의 PSA 결과에 의해 대상 계통의 불가용도 목표치가 책정될 수 있으나, 본 예제에서는 이러한 범위는 포함하고 있지 않으므로 예제에서는 목표 불가용도를 3.45×10^{-6}으로 임의로 설정하였다.

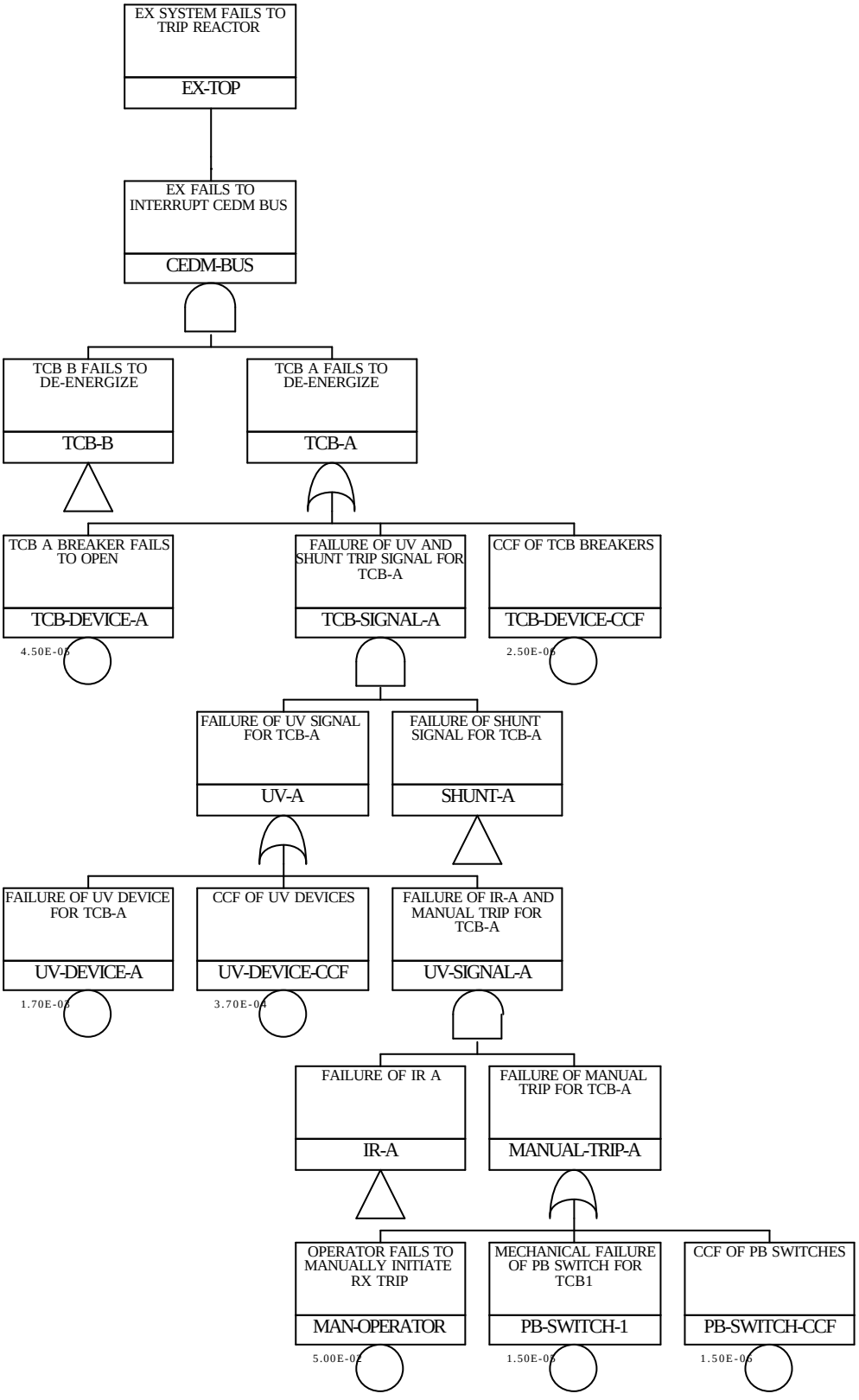
(5) 소프트웨어 모듈의 목표 불가용도 및 사전시험 횟수 계산

<표 2>의 주요 최소단절집합 중 소프트웨어와 관련이 있는 것은 2번에서 5번까지의 최소단절집합들이다. 운전원이 원자로 정지 버튼을 누르는데 실패할 확률을 나타내는 기본사건인 'MAN-OPERATOR'의 확률은 5×10^{-2}이다. 각각의 소프트웨어 고장율이 동일하게 f의 값을 가진다고 가정하면 다음과 같이 f의 값을 계산할 수 있다.

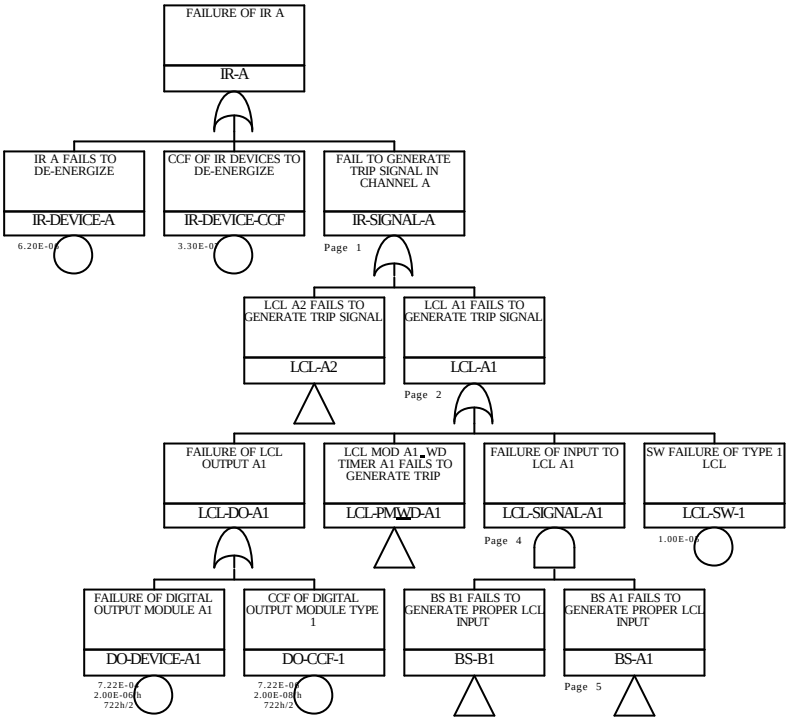
$4\times5\times10^{-2}\times f = 3.45\times10^{-6}-3.42\times10^{-6} = 3.0\times10^{-8}$,

$f = 1.5\times10^{-7}$

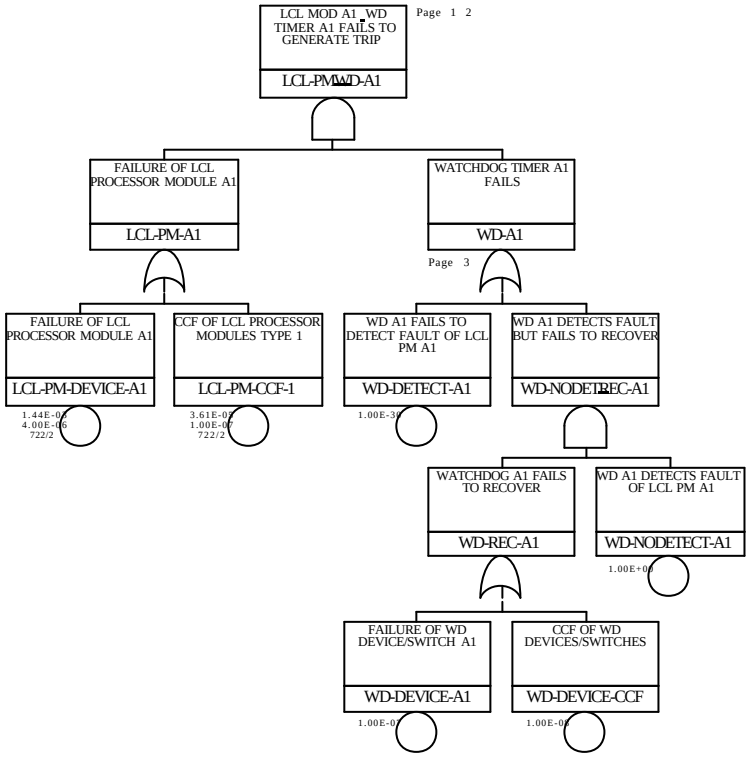
즉, 각 소프트웨어의 고장율이 1.5×10^{-7} 이하의 값을 가지는 것을 사전시험을 통해 보여야 한다. 일반적으로 사용되는 시험횟수 산정방법[7]을 이용하고 확신도를 0.9로 가정하면, 다음과 같은 결과를 얻는다.



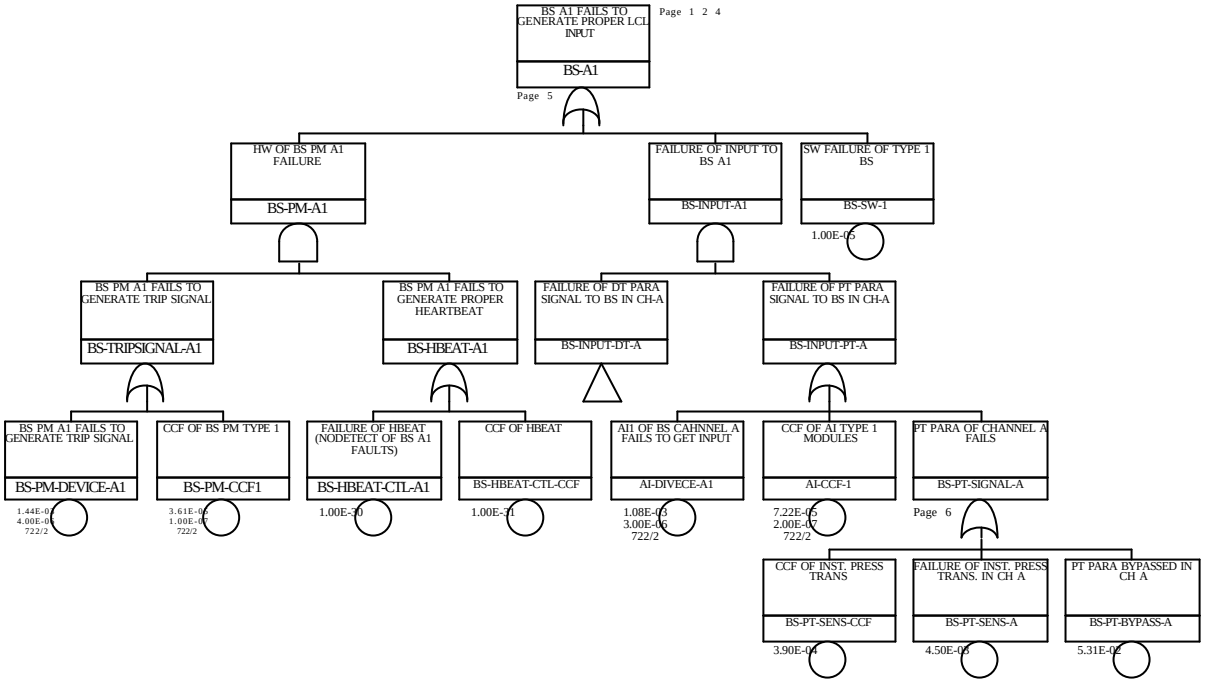
<그림 2> **예제의 보호계통의 고장수목** 1 (Top event)



<그림 3> **예제의 보호계통의 고장수목** 2 (Interposing relay)



<그림 4> **예제의 보호계통의 고장수목** 3 (**동시논리 프로세서와 감시타이머**)



<그림 5> **예제의 보호계통의 고장수목 3 (정지논리 프로세서)**

<표 2> **예제 계통의 주요 최소단절집합**

	확률	기본사건				
1	2.500e-006	TCB-DEVICE-CCF				
2	5.000e-007	BS-SW-1	MAN-OPERATOR			
3	5.000e-007	MAN-OPERATOR	LCL-SW-2			
4	5.000e-007	LCL-SW-1	MAN-OPERATOR			
5	5.000e-007	MAN-OPERATOR	BS-SW-2			
6	3.610e-007	MAN-OPERATOR	DO-CCF-2			
7	3.610e-007	DO-CCF-1	MAN-OPERATOR			
8	2.606e-008	MAN-OPERATOR	DO-DEVICE-A2	DO-DEVICE-B2		
9	2.606e-008	DO-DEVICE-A1	MAN-OPERATOR	DO-DEVICE-B2		
10	2.606e-008	MAN-OPERATOR	DO-DEVICE-A2	DO-DEVICE-B1		
11	2.606e-008	DO-DEVICE-A1	MAN-OPERATOR	DO-DEVICE-B1		
12	1.650e-008	MAN-OPERATOR	IR-DEVICE-CCF			
13	8.510e-009	UV-DEVICE-CCF	SHUNT-DEVICE-CCF			
14	7.605e-009	BS-PT-SENS-CCF	BS-DT-SENS-CCF	MAN-OPERATOR		
15	4.660e-009	BS-PT-BYPASS-A	BS-DT-SENS-CCF	MAN-OPERATOR	BS-PT-SENS-B	
16	4.660e-009	BS-PT-SENS-CCF	BS-DT-BYPASS-A	MAN-OPERATOR	BS-DT-SENS-B	
17	4.660e-009	BS-PT-SENS-CCF	BS-DT-SENS-A	MAN-OPERATOR	BS-DT-BYPASS-B	
18	4.660e-009	BS-PT-SENS-A	BS-DT-SENS-CCF	MAN-OPERATOR	BS-PT-BYPASS-B	
19	2.855e-009	BS-PT-BYPASS-A	BS-DT-BYPASS-A	MAN-OPERATOR	BS-DT-SENS-B	BS-PT-SENS-B
20	2.855e-009	BS-PT-SENS-A	BS-DT-SENS-A	MAN-OPERATOR	BS-DT-BYPASS-B	BS-PT-BYPASS-B
21	2.855e-009	BS-PT-SENS-A	BS-DT-BYPASS-A	MAN-OPERATOR	BS-DT-SENS-B	BS-PT-BYPASS-B
22	2.855e-009	BS-PT-BYPASS-A	BS-DT-SENS-A	MAN-OPERATOR	BS-DT-BYPASS-B	BS-PT-SENS-B
23	2.025e-009	TCB-DEVICE-A	TCB-DEVICE-B			
24	1.625e-009	TCB-DEVICE-A	MAN-OPERATOR	DO-DEVICE-B1		
25	1.625e-009	MAN-OPERATOR	DO-DEVICE-A2	TCB-DEVICE-B		
26	1.625e-009	TCB-DEVICE-A	MAN-OPERATOR	DO-DEVICE-B2		
27	1.625e-009	DO-DEVICE-A1	MAN-OPERATOR	TCB-DEVICE-B		
28	1.408e-009	BS-PT-SENS-CCF	MAN-OPERATOR	AI-CCF-2		
29	1.408e-009	BS-DT-SENS-CCF	MAN-OPERATOR	AI-CCF-1		
30	1.118e-009	BS-DT-SENS-CCF	MAN-OPERATOR	AI-DIVECE-A1	BS-PT-BYPASS-B	
31	1.118e-009	BS-PT-SENS-CCF	BS-DT-BYPASS-A	MAN-OPERATOR	AI-DIVECE-B2	
32	1.118e-009	BS-PT-SENS-CCF	MAN-OPERATOR	AI-DIVECE-A2	BS-DT-BYPASS-B	
33	1.118e-009	BS-PT-BYPASS-A	BS-DT-SENS-CCF	MAN-OPERATOR	AI-DIVECE-B1	

$$n = \frac{\log(1-c)}{\log(1-f)} = \frac{\log(1-0.9)}{\log(1-1.5\times10^{-7})} \approx 1.535\times10^{7}$$

즉, 고전적인 확률이론에 따르면 1.535×10^{7}번 이상의 사전시험을 오류없이 통과해야만 '추후 이 소프트웨어가 실제 사용시 고장을 발생시킬 기대치'를 1.5×10^{-7} 이하로 볼 수 있는 것이다. 이것은 초당 1 회의 시험을 수행한다고 하더라도 180 일이 소요된다는 의미이다.

5. 결론

본 논문에서는 전체 시스템의 확률론적 안전평가(PSA) 결과로부터 소프트웨어가 만족해야 할 불가용도 수준을 추정하고, 이를 이용하여 사전시험 횟수를 결정하는 방법을 제안하고 예제를 통해 설명하였다. PSA 를 이용하여 소프트웨어가 전체 시스템의 신뢰도에 미치는 영향을 체계적으로 분석할 수 있으므로, 적절한 수준의 목표 불가용도를 설정하고 그에 해당하는 사전시험을 수행함으로써 소프트웨어의 건전성을 합리적으로 검증할 수 있을 것으로 판단된다.

사전시험만으로 소프트웨어의 고장율이 결정될 수는 없으나, 정성적인 분석의 결과를 정량화할 방법이 현재까지 정립되어 있지 못하므로 논외로 처리하였다. 추후 이에 대한 보완 연구가 필수적인 것으로 판단된다. 또한 예제에 보인 바와 같이 고전적인 확률이론에 따라 테스트횟수를 산정할 경우 테스트 기간이 비현실적으로 길어진다는 단점이 있으므로 이에 대한 보완 연구도 필요할 것이다.

본 논문의 초점은 고장수목 모델링 방법론이 아니므로 편의를 위해 많은 가정에 기초를 두고 예제 계통의 고장수목을 작성하였다. 이때 이용된 가정들의 정당성은 추후 검증이 필요한 부분이다. 특히 '감시타이머가 동시논리 프로세서에서 발생하는 하드웨어적 오류를 모두 검출할 수 있다'는 가정은 검증이 불가능할 것으로 판단되며 적절한 수준에서 고장 검출률을 추정하여야 할 것으로 판단된다. 그리고 '동일 기종 프로세서의 소프트웨어에만 공통원인 고장이 작용'의 가정도 여러 가지 측면에서 검토가 필요한 것으로 판단된다.

참고문헌

[1] NEA/CSNI/R(97)23, Operating and maintenance experience with computer- based systems in nuclear power plants, 1998.

[2] J.C. Laplace & M. Brun, "Critical software for nuclear reactors: 11 years of field experience analysis," Proceedings of the 9th international symposium on software reliability engineering, p.364-368, 1998.

[3] National Research Council, Digital Instrumentation and Control Systems in Nuclear Power Plant: Safety and Reliability Issues, Chapter 6. Safety and Reliability Assessment Methods, National Academy Press, 1997.

[4] KAERI/AR-571/2000, 디지털 계측 제어 기기의 확률론적 안전성평가를 위한 하드웨어 신뢰도 예측 방법, 2000.

[5] KAERI/AR-565/2000, 소프트웨어 신뢰도의 정량적 평가 기법에 대한 고유현안 분석, 2000.

[6] KAERI/AR-560/2000, 확률론적 안전성 평가에서의 디지털 계측제어 계통 고유현안 분석, 2000.

[7] P.E. Ammann, S.S. Brilliant & J.C. Knight, "The effect of imperfect error detection on reliability assessment via life testing," IEEE Transaction on Software Engineering, Vol. 20, No. 2, 1994.