



KR0000239

KAERI/AR-560/2000

확률론적 안전성 평가에서의  
디지털 계측제어 계통 고유 현안 분석

A Technical Survey on Issues of  
the PSA of Digital I&C Systems

한국원자력연구소

31 / 40

**Please be aware that all of the Missing Pages in this document were  
originally blank pages**

## 제 출 문

한국원자력연구소장 귀하

본 보고서를 2000년도 “차세대원자로 설계관련 요소기술 개발” 과제의 기술현황분석보고서로 제출합니다.

2000. 2. 11.

주 저 자 : 강현국 (종합안전평가팀)

공 저 자 : 성태용 (종합안전평가팀)

엄홍섭 (종합안전평가팀)

정환성 (하나로운영팀)

박진희 (종합안전평가팀)

박진균 (종합안전평가팀)

이기영 (동력로기술개발팀)

박종균 (동력로기술개발팀)

## 요 약 문

원자력 분야에서는 일반 산업계와 비교하여 안전성에 초점을 두고 보수적인 기기 선택 및 적용을 하여 왔으므로, 디지털 기기의 안전 관련계통에 대한 적용 여부에 신중한 자세를 견지해 왔으나, 최근에는 기존 아날로그 보호계통 기기들의 성능저하, 노후화, 부품 품귀 등의 이유로 인해 안전관련 계통에 대해서도 디지털 기기의 도입이 시도되고 있다. 디지털 기기가 단순화, 표준화, 운영, 유지보수 측면에서 많은 장점이 있음에도 불구하고, 원전에서의 적용에 대해 각국의 안전 규제 기관이 신중하고 유보적인 입장을 견지하고 있는 이유 중의 하나는 소프트웨어를 포함하는 디지털 기기에 대한 정량적인 신뢰도 평가 방법이 확보되어 있지 않다는 것이다. 따라서 현재는 소프트웨어 부분에 대해서는 엄격한 명세서와 그 이행사항을 반영한 고품질의 개발 공정을 채택했는지의 여부를 중심으로 정성적인 평가만이 가능하며, 최종 제품의 필수 기능을 검증하는 정도의 실제 시험만이 이루어지는 실정이며, 하드웨어의 부분에 대해서는 기존의 아날로그 하드웨어와 동일하게 취급하고 있다.

따라서 원자력 분야에서 기존에 이용되던 정량적 안전성 평가 방법과 통합하여 사용할 수 있는 디지털 계측제어 시스템의 정량 평가 방법론의 개발이 시급하다. 디지털 시스템의 정량적 안전성 평가에는 기존의 아날로그 시스템과 크게 다른 방법론을 적용하여야 할 것으로 판단된다. 그 이유는 첫째, 디지털 기기에서는 하드웨어의 무작위성 결함 이외에도 소프트웨어 설계 결함으로 인한 결정론적 결함도 고려해야 한다. 둘째, 범용 하드웨어 시스템에 소프트웨어를 통해 기능을 부여하는 방식이므로 특정 기기가 주어진 단일 기능을 수행하는 아날로그와 달라졌으므로 다기능 시스템과 공통의 범용 하드웨어 적용에 따른 공통원인 고장을 고려해야 한다. 셋째, 디지털 기기의 특성상 병렬 배치나 voting 이외에도 다양한 고장내구성 기능의 구현이 가능해 졌으므로, 이를 안전성 평가에 고려해야 한다.

본 보고서는 국내외의 문헌을 통해 디지털 계측제어 계통의 정량적 안전성 분석과 관련된 주요 쟁점들을 다음과 같이 3가지로 정리하여 분석하였다.

1. 소프트웨어와 하드웨어를 분리하여 신뢰도를 추정할 수 있는가 또는 통합

하여 추정하여야 하는가에 관한 문제

하드웨어와 소프트웨어가 동시에 포함된 '시스템'을 평가하기 위한 방법론들이 활발히 제안되고 있었으며, 대체로 하드웨어에 대해서는 무작위성 고장에 중점을 두어 평가하고, 소프트웨어에 대해서는 설계 및 코딩 결함을 주로 다루게 된다.

2. 디지털 계통내의 고장내구성을 확보하기 위해 적용된 다양한 메카니즘을 어떻게 신뢰도 분석에 반영할 것인가에 관한 문제

다양한 계층(소프트웨어, 하드웨어, 시스템)에 다양한 방법으로 적용되고 있는 고장내구성 기법들을 일반적으로 반영할 수 있는 방법론은 아직 개발되어 있지 않은 것으로 파악되었다. 각각의 기법에 대한 확률적 안전평가 (Probabilistic safety assessment; PSA) 방법론이 개별적으로 연구되고 있다.

3. 단일 계통이 다중의 기능을 수행하는 경우의 신뢰도 분석에 관한 문제

마코프 모델이 이러한 시스템을 표현하는데 가장 적합한 것으로 파악되었으나, 마코프 모델의 높은 복잡도와 낮은 적용성이 실제 활용에 장애요인이 되고 있었다. 이를 개선하기 위한 대체 방법론에 관한 연구들이 진행되고 있다.

# 목 차

요약문	2
목차	4
제 1 장 서론	8
제 2 장 소프트웨어와 하드웨어를 포함하는 PSA 방법론	11
제 1 절 개요	11
제 2 절 하드웨어 정량평가 방법론	14
제 3 절 소프트웨어 정량평가 방법론	17
제 4 절 시스템(소프트웨어와 하드웨어) 정량평가 방법론	19
제 5 절 결정론적 방법과 확률론적 방법	28
제 3 장 디지털 계측제어 기기의 고장내구성 기능에 대한 PSA 방법론	30
제 1 절 개요	30
제 2 절 감시 타이머 및 자기검사 기법들	33
제 3 절 고장내구성 소프트웨어	38
제 4 절 시스템 다중성	43
제 4 장 단계화 임무 시스템에 대한 분석 방법론	48
제 1 절 개요	48
제 2 절 BDD를 이용한 단계화 임무 시스템 분석	49
제 3 절 이산 사건 시뮬레이션을 이용한 단계화 임무 시스템 분석	52
제 5 장 결론 및 활용 분야	54

<b>제 6 장</b>	<b>참고 문헌</b>	57
<b>부록 A</b>	<b>COOPRA Digital I&amp;C Working Document 요약</b>	61
제 1 절	디지털 계측제어 시스템 평가 결과의 확률론적 안전성 평가에의 적용	61
제 2 절	PRA의 framework과 계측제어 시스템에 대한 접근 방법	61
제 3 절	정성적 모델의 요건 (Qualitative modeling requirements)	62
제 4 절	정량적 모델의 요건 (Quantitative modeling requirements)	62
제 5 절	디지털 계측제어 시스템 성능 분석의 이슈	64
<b>부록 B</b>	<b>National Research Council의 원전 I&amp;C 보고서 6장 번역</b>	67
제 1 절	서론	67
제 2 절	현재의 US NRC 규제입장과 계획	72
제 3 절	미국 원자력 산업 동향	73
제 4 절	외국의 원자력 산업 동향	73
제 5 절	다른 안전성 중요 산업에서의 동향	74
제 6 절	분석	74
제 7 절	결론 및 권고	75

## 그림 목 차

<그림 2-1> 예제: Tang 등의 방법론을 원자력 발전소 보호 계통에 적용한 경우 .....	21
<그림 2-2> 예제: 3 개의 하드웨어와 단일 소프트웨어로 구성된 시스템에 대한 통합 모델 .....	23
<그림 2-3> 예제: ACIS에 대한 간략화된 고장 수목 .....	25
<그림 2-4> 예제: 프랑스 EDF의 계측제어 기기 선정 과정에 대한 BBN 최상위 모델 .....	27
<그림 3-1> PLC의 작업 수행 시간 개념도 .....	33
<그림 3-2> 감시 프로세서 .....	35
<그림 3-3> n개의 감시 프로세서를 가진 시스템에 대한 모델 .....	36
<그림 3-4> n개의 감시 프로세서를 가진 시스템의 신뢰도 .....	38
<그림 3-5> 1개의 다중성 모듈을 가지는 복구 블록 소프트웨어에 대한 마코프 모델과 계산 결과 예시 .....	42
<그림 3-6> 2 unit warm-standby 시스템의 마코프 모델 .....	45
<그림 3-7> 다중보호(redundant protective) 시스템의 모델 .....	46
<그림 3-8> 보호 시스템의 inspection interval을 결정하기 위한 Anderson의 예제 .....	47
<그림 4-1> BDD를 이용한 우주선의 신뢰도 모델링 .....	50
<그림 4-2> 이산 사건 시뮬레이션을 이용한 terminating application의 모델링을 위한 control-flow graph .....	53



## 표 목 차

<표 2-1> 시스템 신뢰도 분석 기법들의 비교 .....	16
<표 2-2> 소프트웨어 신뢰도 평가 모델 .....	18
<표 3-1> 디지털 시스템의 하드웨어와 소프트웨어에서의 신뢰성 향상 기법 .....	39

## 제 1 장 서론

일반 산업계에는 오래 전부터 마이크로프로세서(컴퓨터)를 이용한 디지털 계측제어 시스템이 보급되어 적극적으로 활용되어 오고 있다. 아날로그 기기에 비해 데이터의 전송과 처리 능력이 뛰어나며, 보다 정확하고 신뢰성있게 신호를 처리할 수 있다는 장점 때문에 디지털 기기들의 사용이 급속히 확산되어 기존의 아날로그 기기를 거의 완전히 대체하고 있다. 그러나 원자력 분야에서는 일반 산업계와 비교하여 안전성에 초점을 두고 보수적인 기기 선택 및 적용을 하여 왔으므로, 최근까지 디지털 기기의 안전 관련계통에 대한 적용 여부에 신중한 자세를 견지해 왔다.

비안전계통의 경우에는 원자력 발전소에도 이미 디지털 기기들이 도입되어 활용되고 있다. 또한 최근에는 기존 아날로그 보호계통 기기들의 성능저하, 노후화, 부품 품귀 등의 이유로 인해, 극히 보수적으로 취급되어 온 안전관련 계통에서도 디지털 기기의 도입이 시도되고 있다. 국내의 경우, 중수로인 월성 원전에서는 안전정지 계통에 이미 디지털 기기들이 도입하여 활용하고 있으며, 경수로에서도 영광 3,4 호기에 마이크로프로세서를 활용한 Interposing Logic System을 도입함으로써 안전 관련 계통에의 디지털 기기의 적용이 시작되었다.

전세계적으로 연구·설계중인 차세대 원자력 발전소의 중요한 특징 중의 하나는 디지털 계측제어계통의 전면 채택이라고 할 수 있다. 디지털 계통 설계의 채택이 단순화와 표준화를 용이하게 하며 운전중 유지보수 측면에서도 많은 장점을 지닌 것으로 판단되기 때문이다. 그러나 이러한 장점에도 불구하고 디지털 계측제어계통 설계에 대해 미국 등 세계 각국의 안전규제 기관의 견해가 신중하고 유보적인 입장을 견지하고 있어 이러한 설계의 실제 채택에 어려움으로 작용하고 있다.

안전 규제기관이 이렇게 유보적인 입장을 보이는 주된 이유 중의 하나는 소프트웨어를 포함하는 디지털 기기에 대한 정량적인 신뢰도 평가 방법이 확보되어 있지 않다는 점이다. 아날로그 기기의 경우에는 정해진 사용구간(입력 범위)내에서 연속적인 거동을 보이므로, 몇 개의 입력 샘플에 대한 결과값을 이용하여 기기의 신뢰도를 추정하는 것이 가능하다. 디지털 기기는 아날로그 기기와는 달리 그 특성이 연속적이지 않다는 점에 정량적 평가의 어려움이 있다. 즉, 디지털 기기는 한정된 샘플 시험값들만으로는 전 사용구간에서의 성능을 추론할 수가 없다는 것이다. 따라서 현재로는 소프트웨어의 부분에 대해서는 엄격한 명세서와

그 이행사항을 반영한 고품질의 개발 공정을 채택했는지의 여부를 중심으로 정성적인 평가만이 가능하며, 최종 제품의 필수 기능을 검증하는 정도의 실제 시험만이 이루어지는 것이 현실이다.

디지털 시스템에 대해서는 기존의 아날로그 시스템에 적용하던 평가 방법과는 크게 다른 방법론의 적용이 요구되고 있다. 범용 하드웨어 시스템에 소프트웨어를 통해 기능을 부여하는 방식부터 아날로그 시스템(설계시 주어진 기능만을 수행하는)과는 크게 다르다. 아날로그 기기에서는 무작위성 결함(random fault)이 주요 결함 요인이었지만, 디지털 기기에서는 하드웨어의 무작위성 결함 이외에도 소프트웨어 설계 결함으로 인한 결정론적 결함(deterministic fault)까지도 고려해야 할 필요가 있다. 소프트웨어 설계의 결정론적 결함은 시험을 통해 완전히 제거하는 것이 불가능하다는 것이 소프트웨어 공학 연구의 대체적인 결론이기 때문이다.

또한 공통의 범용 하드웨어를 사용하므로 공통원인 고장(common cause failure)에 대한 우려가 높아졌다. 소프트웨어의 경우도 코드의 공유나 데이터의 공유가 활발해지므로써 공통원인 고장의 우려를 높이고 있다. 이러한 공통원인 고장의 영향은 원자력 발전소의 안전을 위해 필수적인 다중성의 확보를 저해할 가능성을 높이므로 이를 방지하기 위해 다양성의 확보와 함께 기기의 고품질에 대한 보증이 요구되고 있다. 그러나 전술한 바와 같이 기기의 고품질을 보증할 수 있는 정량적 방법론이 개발되어 있지 않은 상태이므로 이러한 문제에 효과적으로 대응할 수 없다.

이러한 디지털 계통의 정량적 평가 방법론 개발에 대한 시급한 필요성에 의해 본 연구가 수행되고 있는데, 연구 수행의 초기 단계에서 기존의 아날로그 계통과는 크게 달라진 디지털 계통의 특성을 '정량적 평가를 수행하기 위한 관점'에서 분석을 수행하여야 한다. 디지털 계통은 소프트웨어를 포함한다는 점에서 종래의 아날로그 계통과 근본적으로 다르다. 단순히 소프트웨어 자체의 신뢰성 추정이 어렵다는 점만이 달라진 것이 아니라, 소프트웨어와 하드웨어가 맞물려 동작하는 과정에서 기존에 생각하지 못했던 새로운 상황이 많이 나타나기 때문에 이러한 다양한 경우를 정량적 신뢰도 평가에 반영하기 위해서는 새로운 시각으로 접근해야 하는 경우가 많아진 것이다. 이와 관련한 부분에 대해서는 국내외를 막론하고 현재 연구가 진행중인 경우가 많고, 일부 사항에 대해서는 논란이 거듭되고 있는 실정이므로 이를 체계적으로 정리할 필요가 있다.

본 보고서는 국내외의 문헌을 통해 디지털 계측제어 계통의 정량적 안전성 분

석과 관련된 주요 쟁점들을 3가지로 정리하여 분석하였다. 첫번째는 소프트웨어와 하드웨어를 분리하여 신뢰도를 추정할 수 있는가 또는 통합하여 추정하여야 하는가에 관한 문제이다. 두번째는 디지털 계통내의 고장내구성(fault tolerance)을 확보하기 위한 메커니즘을 어떻게 신뢰도 분석에 반영할 것인가에 관한 문제이다. 세번째는 단일 계통이 다중의 기능을 수행하는 경우의 신뢰도 분석에 관한 문제이다. 2장부터 4장에 걸쳐서 전술한 3가지 주제를 각각 다루었다.

## 제 2 장    소프트웨어와 하드웨어를 포함하는 PSA 방법론

디지털 시스템은 하드웨어와 소프트웨어의 결합으로 이루어져 있다. 그런데 이 두 부분의 고장 양상이 크게 다르다는 점이 디지털 시스템의 분석을 어렵게 하는 요인이 되고 있다. 현재의 디지털 시스템에서는 하드웨어와 소프트웨어가 밀접한 관계를 가지고 결합하여 있고 그 수행 기능상의 복잡도가 높아서, 하드웨어와 소프트웨어 각각의 고장율을 추정할 수 있다고 하더라도 이를 독립적으로 반영하여 시스템의 신뢰도를 추정할 수가 없다. 즉, 시스템의 차원에서 하드웨어와 소프트웨어를 동시에 고려하는 모델이 필요하게 된 것이다. 최근 이와 관련한 연구가 국내외에서 진행중이므로 이에 대해 고찰·정리하였다. 2절과 3절에서는 하드웨어와 소프트웨어의 신뢰도 정량평가 방법론을 각각 다루었고, 4절에서 시스템 차원의 접근에 대해 다루었다. 그리고 5절에서는 결정론적 방법과 확률론적 방법의 비교와 최근의 연구동향을 다루었다.

### 제 1 절    개요

하드웨어의 고장율(failure rate)을 정량화하는 방법론이 이미 잘 설정되어 있는 것과는 달리, 소프트웨어 오류에 의해 유발되는 고장율을 추정하는 방법에 관해서는 많은 논란이 있어왔다. 이러한 논란이 현재까지도 계속되고 있어 소프트웨어의 고장율을 추산할 수 있는 일반적인 방법론은 아직 정립되지 못하고 있다. 그러므로 소프트웨어와 관련한 문제는 무시하고 기존의 하드웨어 오류만을 고려하여 디지털 시스템의 신뢰도를 추정하려는 시도들이 있다[1], [2], [3]. 널리 이용되고 있는 표준인 MIL-HDBK-217F, Bellcore Standard TR-332 등의 기존 방법론들도 하드웨어의 부분만을 다루고 있다[4], [5]. 이는 하드웨어의 신뢰도에 관해서는 통계적으로 유용한 자료들을 기존의 많은 응용사례(application)들로부터 쉽게 얻을 수 있다는 점에 기인한다. 소프트웨어나 인간 운전원(human operator)에 의한 시스템의 고장 유발에 관련한 연구는 상대적으로 그 역사가 짧다. 따라서 분석에 필요한 자료의 수가 부족하며, 통계적 처리 방법에 관해 통일된 의견이 없는 상태이다. 또한 소프트웨어의 특성상 과연 통계적으로 다룰 수 있는 문제인지 자체가 논란의 대상이 되고 있어, 일반적으로 공감하는 평가방법론이 완

전히 정착되기까지는 많은 시간이 걸릴 것으로 판단된다.

그러나 최근에 소프트웨어의 오류가 전체 시스템 고장의 중요한 요인이 된다는 연구 결과들이 발표되면서, 하드웨어만으로 전체 시스템의 고장율을 예측하려는 시도는 점차 그 입지가 좁아지고 있다. 미국의 원자력 발전소 운전 경험에 의하면 디지털 시스템의 고장 중 38% 정도가 소프트웨어의 문제에 기인하는 것으로 밝혀졌으며, 캐나다의 경우 35% 정도가 소프트웨어의 문제에 기인하는 것으로 보고되었다 [6]. 즉, 소프트웨어를 제외한 하드웨어만으로 신뢰도를 계산하였을 경우, 고장율에 대해 약 3분의 1가량 저평가된 결과를 얻게 된다는 것이다. 또한 소프트웨어의 오류로 인한 고장의 경우 그 수리 시간(repair time)이 디지털 하드웨어의 경우보다 긴 것이 일반적이므로, 불가용도(unavailability) 측면에서는 훨씬 더 큰 오차를 일으키게 될 것으로 판단된다.

기존에는 소프트웨어와 하드웨어의 신뢰성에 대해 각각의 연구가 독립적으로 진행되어 왔으므로, 이들을 별개로 취급하여 각각의 고장율을 각각의 방법으로 추정한 후, 이 두가지 값을 더하여 전체 시스템의 고장율을 추정하려는 연구들이 있다 [7], [8]. 그러나 이는 한가지의 기능(function)이 특정한 한 시스템에서 수행되는 것을 전제로 한 경우에만 성립하는 것으로써, 현재의 복잡한 시스템의 경우에는 하나의 시스템이 여러 가지의 기능을 수행하거나 여러 개의 시스템이 한가지 기능을 수행하는 경우가 많고, 자기 검사(self-testing)와 같이 신뢰성과 깊은 관련이 있으나 시스템의 고유기능 수행과는 관련이 없는 부분이 많아져서, 소프트웨어와 하드웨어의 고장률을 독립적으로 추정하여 더한 값이 시스템의 신뢰도를 적절히 대변하지 못하는 경우가 많다. 즉, 기능과 시스템을 동일한 개체로 취급할 수 없게 된 것이다. 또한 소프트웨어와 하드웨어를 별도로 분석할 경우, 두 개체사이의 상호작용을 무시하게 되어 결함 masking과 같은 효과를 고려할 수 없게 된다 [9].

이에 따라, 하드웨어와 소프트웨어가 동시에 포함된 '시스템'을 고려하기 위한 방법론이 제안되고 있다 [10], [11], [12], [13], [14], [15], [16]. Hecht와 Tang의 연구는 최종 시험 단계나 사용 단계에서 시스템의 신뢰도를 추정하는 방법에 관한 것이다. 먼저 시스템에 대한 마코프 모델을 만들고 실험에 의해 측정된 결과로부터 모수(parameter)를 추정하는 방식이다. 마코프 모델에서는 소프트웨어나 하드웨어의 설계 오류 등 결정론적인 오류에 의해 시스템이 적절히 반응하지 못하는 경우와 하드웨어에 발생하는 무작위성 결함에 의한 고장의 경우를 구분하여 천이 확률을 구한다. 그러나 이 방법은 실측 실험값(test data)에 의존하여 신

뢰도를 추산하는 방식이기 때문에, 고신뢰도( $10^{-6}$  #/hr 이하) 시스템에는 적용하기 어렵다. 또한 이러한 순수한 확률적 접근 방법은 대상 시스템의 고장 양상이 stochastic process와 얼마나 근접한 지에 의해 신뢰도 추정 가능성이 크게 좌우되므로, 그 분포를 추정하기 어려운 경우에는 만들어진 모델이 부적절한 계산 결과를 제공할 가능성이 높다.

Welke 등의 연구결과는 마코프 모델을 이용하여 하드웨어와 소프트웨어가 포함된 시스템을 모사한다는 점에서 Tang 등의 연구와 유사하나, 소프트웨어에 대해서 fault-counting 방법을 적용하여 고장율( $\lambda$  값)을 직접 구하여 적용한다는 점이 다르다. 즉, Tang 등은 '전체 시스템이 특정 기능 수행모드에 성공적으로 들어갈 확률'에 소프트웨어와 하드웨어의 설계 오류에 의한 고장을 모두 포함시켰지만, Welke 등은 소프트웨어 자체의 고장율을 따로 구해서 소프트웨어와 하드웨어의 상호작용(interaction)을 직접적으로 고려할 수 있도록 하였다. 이러한 방법으로 소프트웨어의  $\lambda$  값을 구하는 것이 적절한지에 대해서는 아직 논란이 많은 상태이므로 이 부분에 취약점이 있다고 하겠다. 또한 소프트웨어와 하드웨어가 상호작용을 하여 기능을 수행하는 과정을 직접 모델링해야 하므로 분석자의 주관이 개입할 여지가 많으므로 주관의 개입을 최소화할 수 있는 적절한 분석틀(framework)이 필수적이다.

한편, Vemuri와 Kaufman 등이 제안하는 방법은 고장 수목(fault tree)을 이용하여 시스템의 신뢰도를 추정하려는 시도이다. 고장 수목을 이용하여 고장 양태에 대해 분석적으로 접근하기 때문에, 시스템 전체에 대한 실험 데이터에만 의존하는 Hecht와 Tang의 연구와는 차별될 수 있다. 이들은 하드웨어와 소프트웨어 사이에 기능적 종속성(functional dependency)이 존재한다는 점을 파악하고, 이러한 관계를 고장 수목으로 표현하였다. 이렇게 작성된 고장 수목은 대단히 복잡하고 모듈화하기 어렵기 때문에, 그대로 풀 수가 없고 마코프 사슬 모형으로 옮겨서 풀게 된다. 이때 고장 수목을 마코프 사슬 모형으로 옮겨주는 과정은 자동화될 수 있다. 그러나 소프트웨어의 고장율을 추정하여야 한다는 점이 여전히 문제로 남아 있다.

Bouissou 등의 방법은 Bayesian belief network (BBN)을 이용하여 소프트웨어를 포함한 디지털 시스템을 평가하려는 시도이다. 기존의 방법으로는 소프트웨어에 내재되어 있는 설계 오류를 평가하기 어려웠던 것에 비해서 이 방법은 전문가들의 의견을 효율적·정량적으로 반영할 수 있다는 장점이 있다. 그러나 전문가의 지식을 정량화하여 BBN으로 표현하는 과정에서 일관성과 타당성을 확보

하는 것이 필요하다.

근래의 연구동향은 대체로 하드웨어와 소프트웨어를 포함한 전체 시스템의 구조와 기능을 정리하고 그 고장 유형(failure mode)에 대한 분석을 수행한 후, 신뢰도 모델링(reliability modeling)을 수행하는 방향으로 정리되고 있는 듯하다. 이때 하드웨어에 대해서는 무작위성 고장에 의한 고장율을 고려하여 주어야 하며, 소프트웨어에 대해서는 설계 오류와 코딩 오류에 의한 고장을 주로 고려하게 된다. 이렇게 시스템의 여러 가지 측면을 생명 주기(life cycle)까지 고려하여 분석하고자 하는 동향을 단계별 접근법(Layered approach)라고 하는데, Karydas와 Brombacher가 정리한 단계별 접근법의 적용 순서와 개요는 다음과 같이 나타낼 수 있다 [17].

- 1) 시스템 구조 모델링(architecture modeling)
- 2) 하드웨어 고장 유형 및 확률(modes & probabilities)분석
- 3) 시스템 전체의 고장 유형을 분석(systemic failure mode analysis)
- 4) 신뢰도 모델링(reliability modeling) 수행

## 제 2 절 하드웨어 정량평가 방법론

기존의 아날로그 계측제어 시스템에서는 소프트웨어를 고려할 필요가 없었으므로, 그 신뢰도 추정을 위해 하드웨어의 정량평가만을 수행하면 충분하였다. 하드웨어의 고장은 제품 생산 공정에 근본적인 문제가 있는 경우를 제외하고는 무작위성 고장의 형태로 나타난다. 기존의 아날로그 시스템의 경우, 사용되는 부품의 개수가 많지 않고 종류도 한정되어 있으므로 고장율을 추정하기가 상대적으로 용이하며, 오랜 사용 경험을 가지고 있으므로 그 고장의 확률적 분포가 잘 알려져 있어 시스템의 신뢰도를 비교적 정확하게 추정할 수 있다.

하드웨어의 정량적 평가는 기본이 되는 부품이나 단위 계통의 고장율 값을 먼저 구하고 그 값들을 조합하여 시스템 전체의 신뢰도를 예측할 수 있는 모델에 적용하여 계산하는 과정을 통해 수행된다. 따라서 신뢰도 분석·예측 방법들은 이 모델을 구성하는 기법에 해당하는 것들이다. 이 모델이 보다 정확하게 부품이나 단위 계통의 고장율과 전체 시스템의 신뢰도를 연결시켜 줄수록 보다 유용한 신뢰도 예측이 가능해 지는 것이다. 그러므로, 하드웨어에 대한 정량 평가 방법은 두가지 요소로 이루어진다고 할 수 있는데, 기초 고장율을 추정하는 방법과



모델을 구성하는 방법이 그 요소들이다.

기초 고장율을 추정하는 방법도 테스트를 통해 수집된 자료를 통계적 분포에 적용하여 분석하는 직접적인 방법과 일반적으로 공인 받는 자료들(prediction handbook)을 이용하여 간접적으로 추정하는 방법이 있다. 직접적인 테스트를 통해 고장의 분포를 신뢰성있게 추정하기 위해서는 많은 시간과 노력이 소모되므로 간접적인 추정방법이 보편적으로 이용되고 있는데, 주로 이용되는 방법론으로는 MIL-HDBK-217F, Bellcore Standard TR-332, British Telecom HRD4 등이 있다 [18]. MIL-HDBK-217F는 주로 군수 용품 등 보수적 신뢰도 추정이 필요한 경우에 많이 이용되고 있으며, 상업적인 일반 전자 제품에는 Bellcore Standard TR-332이 널리 이용되고 있다. 일반적으로 미국 이외의 지역에서는 MIL-HDBK-217F의 자료가 가장 많이 이용된다. MIL-HDBK-217F를 이용하여 보다 정확한 고장율을 추정해 내기 위해서는 각 부품의 기저 고장율( $\lambda_b$ ), 품질 조정 요소( $\pi_Q$ ), 환경 조정 요소( $\pi_E$ ), 적용 조정 요소( $\pi_A$ ), 반복 효과( $\pi_n$ ), 온도( $\pi_T$ )의 정보가 필요하다.

모델을 구성하여 신뢰도를 예측하기 위해서 이용되는 대표적인 방법론으로는 MIL-HDBK-217F에서 제시한 부품 스트레스 분석법(part stress analysis)과 부품 카운트 분석법(part count analysis)이 있고, 고장 유형 효과 분석법(FMEA), 신뢰도 블럭도(RBD), 고장 수목 분석법(FTA), 마코프 모델 등이 있다. 엄밀한 의미에서, 부품 스트레스 분석법과 부품 카운트 분석법은 기초 고장율을 추정하는 기법과 모델을 구성하는 기법의 통합체라고 볼 수 있다. 이 기법들을 이용할 경우 전체 시스템의 신뢰도는 추정된 부품의 고장율의 합에 의해 결정된다. 신뢰도 블럭도, 고장 수목 분석법, 마코프 확률 모델 등은 시스템의 기능과 구조를 고려하여 보다 효과적으로 전체 시스템의 신뢰도를 추정할 수 있도록 해 준다. <표 2-1>은 이러한 방법론들의 적용범위와 장단점을 비교하여 정리한 것이다 [19].

<표 2-1> 시스템 신뢰도 분석 기법들의 비교 [19]

Aspects \ Methods	Parts count analysis	Parts stress analysis	Fault tree analysis	Reliability block diagram	Markov analysis
Safety ranking/ comparison	○	○	○	○	○
Probability of dangerous failures	○	○	○	○	○
Availability prediction	○	○	○	○	○
Triprate prediction			○		○
Effects of redundancy			○	○	○
Common cause failures			○	○	○
Systematic failures				○	○
Effect of diagnostics			○		○
Effects of on-line test and repair			○		○
Effects of off-line test and repair			○		○
Time/sequence dependent aspects					○

### 제 3 절 소프트웨어 정량평가 방법론

1960년대까지만해도 컴퓨터 시스템의 하드웨어에 관련된 성능 평가에 연구의 관심이 집중되어 왔으나, 1970년대 이후 하드웨어에 비하여 소프트웨어가 복잡해지고 가격이 급격히 상승함에 따라 소프트웨어 자체에 관심이 기울여지기 시작하였다. 최근에는 원자력계에서도 소프트웨어의 오류가 전체 시스템 고장의 중요한 요인이 된다는 연구 결과들이 발표되고 있다. 미국의 운전 경험에 의하면 디지털 계측제어 시스템의 고장 중 38% 정도가 소프트웨어의 문제에 기인하는 것으로 밝혀졌으며, 캐나다의 경우 35% 정도가 소프트웨어의 문제에 기인하는 것으로 밝혀졌다 [6].

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 과연 통계적으로 다를 수 있는 문제인지 자체가 논란의 대상이 되고 있어, 일반적으로 공감하는 평가방법론이 완전히 정착되기까지는 많은 시간이 걸릴 것으로 판단된다. 이러한 소프트웨어로 인한 고장 분석에 관한 논란이 현재까지도 계속되고 있어 소프트웨어의 고장율을 추산할 수 있는 일반적인 방법론은 아직 정립되지 못하고 있다.

소프트웨어의 정량적 평가는 하드웨어의 경우와는 다른 점이 많다. 가장 큰 차이점은 소프트웨어의 경우 생명주기 전체에 대한 평가가 중요하게 작용한다는 것이다. 하드웨어의 경우에도 철저한 설계 검증과 생산·품질 관리를 거친 제품의 신뢰도가 높다는 점은 마찬가지이나, 하드웨어 고장의 주된 원인이 우발 사건이기 때문에, 완제품에 대한 테스트 결과가 생명주기에 대한 평가보다 더 큰 설득력을 갖는다. 그러나 소프트웨어의 경우 그 주된 고장 원인이 우발 사건이라고 볼 수 없다는 의견이 많다. 그러므로 처음의 설계 단계에서부터 마지막 완제품에 이르기까지의 과정을 평가하는 것이 보다 높은 중요성을 가지게 된다. Laplace 등은 소프트웨어의 경우 개발 과정에 대한 철저한 관리만으로도 충분한 수준의 신뢰도를 확보할 수 있다고 주장하고 있다 [20]. 실제 운전 경험은 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, Laplace등의 주장을 전적으로 따르기는 어려울 것으로 생각된다.

이와 같이 생명주기 관리에 대한 부분을 포함하는 소프트웨어 신뢰도 평가 방법론은 소프트웨어 신뢰도가 각 개발단계에 의해 영향을 받는다는 가정하에, 소프트웨어 명세, 설계, 구현 단계에 대한 품질평가를 통해 소프트웨어의 신뢰도를 간접적으로 예측하는 방법이다. 소프트웨어의 생명주기 관리를 평가하므로써 최종 결과물인 소프트웨어의 신뢰도를 추정해 보려는 연구의 장·단점에 대해서는 본 연구를 통해 발행한 기술현황 분석보고서[21]에 상세히 다루었으므로 이를 참조하기 바란다. 본 보고서에서는 소프트웨어와 하드웨어가 통합된 ‘시스템’의 문제를 확률적으로 다루기 위한 노력을 중심으로 분석을 수행하였으므로, 소프트웨어의 경우에 대해서도 확률적으로 다룰 수 있는 기법들을 중심으로 기술해 나가기로 한다. 소프트웨어의 생명주기 관리정도를 포함하는 신뢰도 계산 기법을 도입하는 경우라도 필요한 부분만 가감하면 확률적 기법의 사용과 유사할 것으로 판단된다.

<표 2-2> 소프트웨어 신뢰도 평가 모델

Time-dependent Model		Time-independent Model	
MTBF Models	Failure Counts Model	Fault Seeding Model	Input-domain Based Model
Jelinski-Moranda Model	Goel-Okumoto NHPP Model	Mill's Seeding Model ...	Nelson Model
Schick-Wolverton Model	Goel Generalized NHPP Model		Ramamoorthy-Bastani Model
Littlewood-Verrall Bayesian Model	Shooman Exponential Model		...
...	...		...

소프트웨어 자체의 신뢰도를 평가하기 위한 노력들이 많은 연구에 의해 행해졌는데, 그것을 분류하여 <표 2-2>와 같이 정리할 수 있다 [22]. 소프트웨어의 신뢰도를 평가하는 모델은 크게 시간 종속적 모델(time-dependent model)과 시

간 독립적 모델(time-independent model)로 나눌 수 있다. 시간 종속적 모델은 소프트웨어가 가동중인 시간에 의존하여 소프트웨어의 고장이 발생한다는 가정 하에 신뢰도를 평가하고자 하는 모델이며, 시간 독립적 모델은 일정한 시험 입력을 이용하여 소프트웨어를 시험한 결과를 근거로 신뢰도를 측정한다는 모델이다. 각 모델에 대한 상세한 설명은 [21]와 [22]에 제시되어 있다.

이렇게 최종 결과물인 '소프트웨어 자체'의 신뢰도를 직접 평가하려는 방법론의 적용상의 문제점은 크게 두가지에 기인한다. 그 첫번째는 자료 취득의 어려움으로, 자료를 취득하기 위한 노력이 과다할 것이라는 것은 물론이고 취득된 자료의 엄밀함을 검증하기도 어렵다. 두번째 문제는 취득된 자료를 통해 고장의 분포를 추정하는 과정에서 나타난다. 지금까지 소프트웨어의 고장을 적절히 설명해주는 것으로 일반적으로 인정되는 확률 모형이 없다는 것이다. 특히 복잡한 소프트웨어의 경우 이 문제를 해결하기가 대단히 어렵다.

이상에서 설명한 바와 같이, 소프트웨어의 확률론적 신뢰도 정량 평가는 대단히 어려운 작업으로서, 몇몇 실험실 차원의 연구결과들을 제외하고는 현재까지는 뚜렷한 결론에 도달하지 못하고 있다. 그러나 하드웨어에서의 경우와 같은 방법론을 적용할 수 없다는 점은 점차 분명해 지고 있는 것으로 보인다. (초기의 소프트웨어 정량평가에 관한 연구들은 하드웨어에 적용하던 방법을 그대로 적용하려는 시도들이 대부분이었다.) 소프트웨어의 정량평가를 위해서는 위에 언급한 부분들 이외에도, 공통 모드 또는 공통 원인 고장에 대한 분석 및 평가를 수행하여야 한다. 즉, 다양성과 다중성의 확보를 증명하기 위한 연구가 수행되어야 할 것으로 판단된다.

## 제 4 절 시스템(소프트웨어와 하드웨어) 정량평가 방법론

전술한 바와 같이 기존에는 소프트웨어와 하드웨어의 평가에 대해 각각의 연구가 독립적으로 진행되어 왔으므로, 이들을 별개로 취급하여 각각의 고장율을 구한 후, 이 값을 더하여 전체 시스템의 고장율을 추정하려는 연구 경향이 있었다 [7], [8]. 그러나 이는 기능(function)과 시스템을 동일한 개체로 취급할 수 있는 경우에만 성립하는 것으로써, 최근의 시스템은 훨씬 복잡해져서, 하나의 시스템이 여러 가지의 기능을 수행하거나 여러 개의 시스템이 한가지 기능을 수행하는 경우가 많으므로 전체에 대한 적절한 모델링없이 독립적으로 추정하여 단순

히 합산하는 것만으로는 시스템의 신뢰도를 추정할 수 없는 경우가 많다. 또한 소프트웨어와 하드웨어를 별도로 분석할 경우, 두 개체사이의 상호작용을 무시하게 되어 fault masking과 같은 효과를 고려할 수 없게 된다 [9].

이에 따라, 하드웨어와 소프트웨어가 동시에 포함된 '시스템'을 고려하기 위한 방법론이 제안되고 있다. 이러한 방법론들이 현재 활발히 연구·개발중인 상태이므로 이 보고서에서 최근의 대표적인 연구 동향을 정리하였다.

## 1. 시험 자료를 이용한 모델

최종 시험 단계나 사용 단계에서 취득한 테스트 자료를 이용하여 시스템의 신뢰도를 추정하는 방법을 Hecht와 Tang이 제안하였다 [10], [11], [12]. 먼저 시스템에 대한 마코프 모델을 만들고 실험에 의해 측정된 결과로부터 모수(parameter)를 추정하는 방식이다. 소프트웨어나 하드웨어의 설계 오류 등 결정론적인 오류에 의해 시스템이 적절히 반응하지 못하는 경우와 하드웨어에 발생하는 무작위성 결함에 의한 고장의 경우를 구분하여 따로 확률을 구하여 적용한다. 즉, 일반적인 모델링의 경우와는 달리, 마코프 모델에서 고장 상태(failure state)로 천이할 확률이 확률론적으로 결정되는 경우와, 결정론적으로 결정되는 경우로 양분하는 것이다. 마코프 모델을 이용하기 때문에 상위 개념 모델에서 하위 상세 모델 단계로 점차 그 영역을 확장할 수 있는 장점이 있다. 또한 수리(repair)를 포함한 모델을 작성하는 것이 가능하며, 시스템이 자기 감시 기능을 가지거나 다중의 백업 시스템을 가지는 경우에 대해서도 유효범위치(coverage factor)를 이용하여 표현이 가능하다.

<그림 2-1>은 Tang 등이 제안한 모델을 설명하기 위한 예제이다. 원자력 발전소 보호 계통에 적용한 경우에 대한 예제인데, 그림에 나타낸 것은 최상위의 간략한 마코프 모델이며, 각각의 state를 보다 상세화해 나갈 수 있다. 먼저 각 state부터 설명하면 다음과 같다.

Sns: 정상 상태 (보호시스템에 작동신호 발생하지 않음),

Ssp: 보호시스템이 작동중인 상태,

Ssf: 보호시스템이 작동하는데 실패한 상태 (작동신호는 발생하지 않았음),

Sph: 원자로가 위험하게 된 상태.

천이 확률에 사용된 기호들은 다음과 같은 의미를 가진다.

Ps: Probability of success upon demand, 작동신호 발생시 보호시스템이 성

공적으로 기동하여 동작 상태가 되는 확률 ('시스템의 기동'에만 국한하며  
 그후의 동작의 결과가 성공적이었는지는  $\lambda_{ss}$ 를 이용하여 표시한다)

$r$ : Arrival rate of challenges from the plant requiring a response of the  
 safety system, 작동신호가 발생하는 빈도

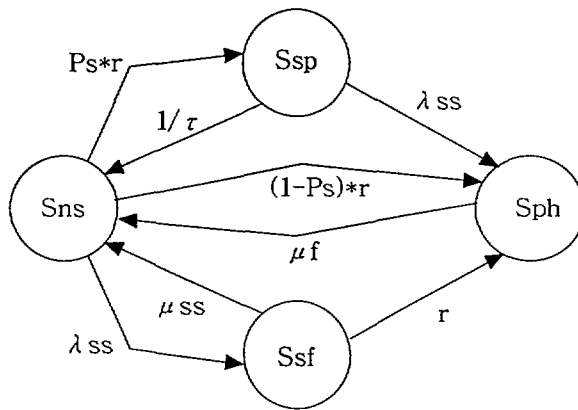
$\lambda_{ss}$ : Failure rate of the safety system, 보호시스템이 작동에 실패할 확률

$\tau$ : Challenge processing time, 보호시스템이 작동신호에 기동하여 정해진 기  
 능을 완수하는데 걸리는 시간

$\mu_{ss}$ : Rate for detection and handling of a safety system failure, 보호시스  
 템의 고장 사실을 감지하여 수리할 확률

$\mu_f$ : Recovery rate of the plant after a hazardous event, 위험한 상태로 갔던  
 시스템이 원상 복귀할 확률

이 모델에서 소프트웨어 등의 결정론적인 고장은  $1-P_s$ 로 표현된다. 즉 어떠한  
 이유로든 (소프트웨어나 하드웨어의 설계 오류 등에 의해) 시스템이 적절히 반응  
 하지 못하는 경우가 될 확률을  $1-P_s$ 로 하는 것이다. 일단 적절한 반응(기동)을  
 시작한 시스템이 우발적 고장에 의해 그 기능을 수행하지 못할 확률(고장율)은  
 $\lambda_{ss}$ 로 표시된다.



<그림 2-1> 예제: Tang 등의 방법론을 원자력 발전소 보호 계통에  
 적용한 경우 (최상위의 마코프 모델)

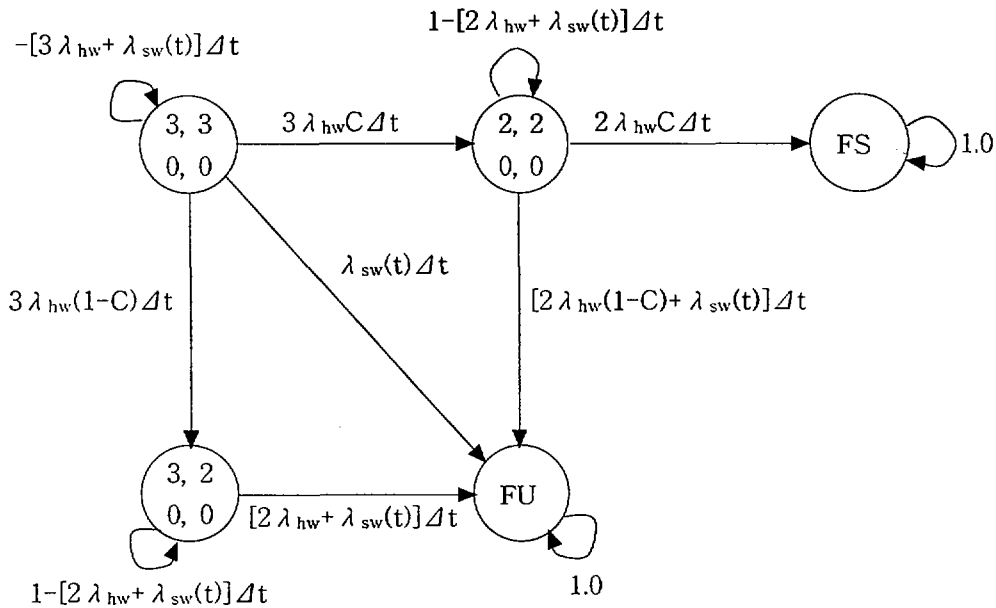
Tang 등의 방법론은 전형적인 시간 종속형 모델이다. 소프트웨어로 인한 고장과 하드웨어로 인한 고장을 분리하지 않고 시스템 고장 상태로 가는 확률을 두가지로 분리하여 모델을 작성하므로써 디지털 시스템의 상태를 효율적으로 표현하고 있다. 그러나 소프트웨어의 부분에 대해 특별한 처리를 해 주는 것이 아니고 하드웨어의 경우와 같이 완전히 시간에 대한 확률 함수로 표현이 가능한 것으로 취급하고 있다는 단점이 있다. 또한 이 방법은 실측 실험값에 의존하여 신뢰도를 추산하는 방식이기 때문에, 고신뢰도( $10^{-6}$  #/hr 이하) 시스템에는 적용하기 어렵다. 원자력 발전소의 안전 관련 시스템과 같이 대상 상황(비정상 운전 상태)이 흔치 않은 경우에는 더욱 적용하기 어렵다. 또한 이러한 순수한 확률적 접근 방법은 대상 시스템의 고장 양상이 stochastic process와 얼마나 근접한 지에 의해 신뢰도 추정 가능성이 크게 좌우되는데, 소프트웨어를 포함하는 시스템의 경우에는 확률적 분포를 추정하기 어려운 경우가 많으므로 실제 적용시에는 상당한 주의가 필요하다.

## 2. Welke의 통합 모델(unified model)

Welke 등은 소프트웨어에 대해서 fault-counting 방법을 적용하여 고장율( $\lambda$  값)을 직접 구하여 하드웨어 신뢰도 모델에 포함시켜 전체 시스템의 신뢰도를 계산하는 방법을 제안하였다 [13]. Welke 등은 이것을 통합 모델(Unified Model)이라고 불렀다. 이 방법론은 마코프 모델을 이용하여 하드웨어와 소프트웨어가 포함된 시스템을 모사한다는 점에서 Tang등의 연구와 유사하나, 소프트웨어의 고장율을 직접 추정한다는 점에서 차이가 있다. 즉, Tang등은 '전체 시스템이 특정 기능 수행모드에 성공적으로 들어갈 확률'에 소프트웨어와 하드웨어의 설계 오류에 의한 고장을 모두 포함시켰지만, Welke등은 소프트웨어 자체의 고장율을 따로 구해서 소프트웨어와 하드웨어의 상호작용(interaction)을 직접적으로 고려할 수 있도록 하였다.

<그림 2-2>는 Welke의 통합 모델 방법론을 설명하기 위해 TMR (Triple modular redundancy) 시스템을 모델링한 예제이다. TMR은 3개의 하드웨어로 구성된 시스템인데, 각 하드웨어에 탑재된 소프트웨어는 동일한 것으로 가정하여 모델링한다. 3개중 2개의 모듈(하드웨어, 소프트웨어 포함)이 정상적으로 작동하여야 주어진 기능을 충족할 수 있는 것으로 가정하며 예비품(spare part)은 없는 것으로 한다.





<그림 2-2> 예제: 3 개의 하드웨어와 단일 소프트웨어로 구성된 시스템에 대한 통합 모델 (2개 이상의 모듈이 정상 작동하여야 시스템의 기능 충족)

먼저 각 state에 대해 설명을 하면 다음과 같다. FS state는 전체 시스템이 기능을 충족하는데 실패하였으되, 사용자가 시스템이 실패하는 것을 알아 챌 수 있는 경우이다. 이 예제에서 이러한 경우는 안전한 실패(fail safe)로 규정한다. 반대로 전체 시스템이 기능 수행에 실패하였을 때, 사용자가 전혀 이를 알아채지 못하고 있는 경우를 안전하지 못한 실패(fail unsafe)로 규정하며, 이를 FU state로 나타낸다. 그 외에는 state의 원안에 4개의 숫자가 기입되어 있는데, 왼쪽 위의 숫자부터 시계방향으로 돌면서 그 의미를 설명하면 다음과 같다.

- 'Fail하지 않은 모듈의 수'와 'fail하였으되 그 failure 여부가 검출되지 않은 모듈의 수'의 합
- 실제로 fail하지 않은 모듈의 수
- 실제로 fail하지 않은 예비품의 수 (이 예제에서는 항상 0)
- 'Fail하지 않은 예비품의 수'와 'fail하였으되 그 failure 여부가 검출되지 않은 예비품의 수'의 합 (이 예제에서는 항상 0)

각 천이확률에 사용된 기호의 의미는 다음과 같다.

$\lambda_{hw}$ : Failure rate of hardware, 하드웨어의 고장율

$\lambda_{sw}$ : Failure rate of software, 소프트웨어의 고장율

C: 고장검출 확률

이 방법은 소프트웨어 자체에 대해 확률 모수인  $\lambda$  값을 직접 구할 수 있는지, 구할 수 있다면 구하는 방법이 적절한지에 대해서 아직 논란이 많은 상태이므로 이 부분에 취약점이 있다고 하겠다. 또한 소프트웨어와 하드웨어가 상호작용을 하여 기능을 수행하는 과정을 직접 모델링을 해야 하므로 분석자의 주관의 개입할 여지가 많으므로 적절한 분석틀(-framework)이 필수적이다.

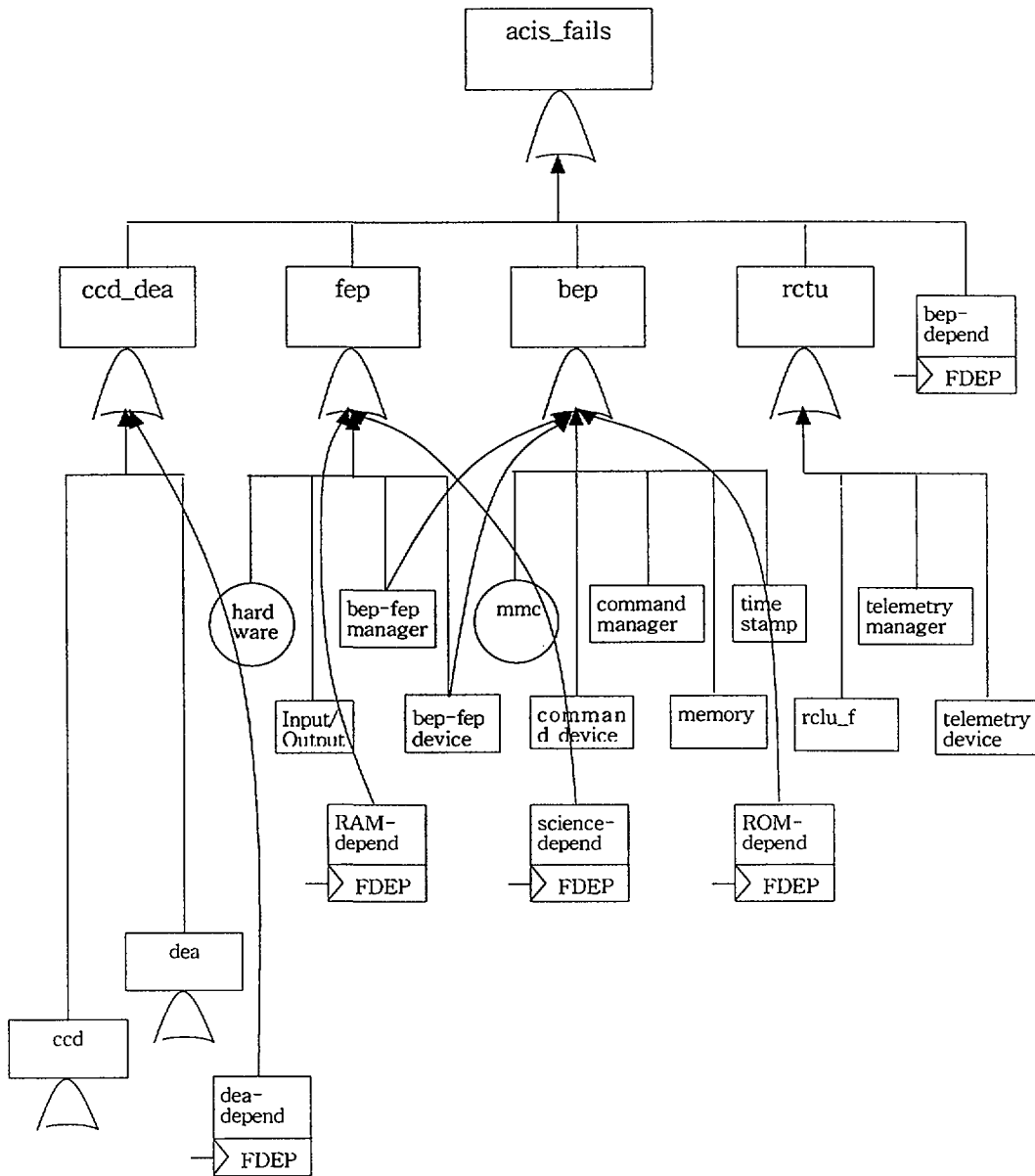
### 3. 고장 수목을 이용한 방법

Vemuri와 Kaufman 등이 제안하는 방법은 고장 수목(fault tree)을 이용하여 시스템의 신뢰도를 추정하려는 시도이다 [14], [15]. 고장 수목을 이용하여 고장 양태에 대해 분석적으로 접근하기 때문에, 시스템 전체에 대한 실험 데이터에만 의존하는 Hecht와 Tang의 연구와는 차별될 수 있다. 이들은 하드웨어와 소프트웨어 사이에 기능적 종속성(functional dependency)이 존재한다는 점을 파악하고, 이러한 관계를 고장 수목으로 표현하였다. 이렇게 작성된 고장 수목은 대단히 복잡하고 모듈화하기 어렵기 때문에, 그대로 풀 수가 없고 마코프 사슬 모형으로 옮겨서 풀게 된다. 이때 고장 수목을 마코프 사슬 모형으로 옮겨주는 과정은 자동화될 수 있다. 이 방법론을 다시 설명하자면, 비교적 정확한 모델링이 가능하다고 알려진 마코프 모델을 이용하여 하드웨어와 소프트웨어가 서로 영향을 미치는 시스템을 표현하되, 처음부터 직접 마코프 모델으로 이러한 관계를 모두 표현하는 것은 대단히 복잡한 작업이 되므로, 중간에 고장 수목을 이용하여 실제 시스템과 마코프 모델 사이의 매개로 삼는 것이다.

[14]에 제시된 예제를 살펴보면 다음과 같다. MIT 우주 센터에서 개발된 ASIC이라는 계측장비에 대한 신뢰도 해석을 수행하였는데, ASIC은 CCD image spectrometer, digital image processor 등의 복잡한 장비를 포함하며, 이러한 장비들을 운용하여 원하는 목적을 달성하기 위한 소프트웨어들이 실려있다. 고장 수목을 작성하기 위해서는 먼저, 시스템의 기능, 다중 component가 있는지 여부, failure events, covered events 등을 파악하여야 한다. 하드웨어와 소프트웨어가 포함된 복잡한 시스템을 고장 수목으로 표현하는 것은 힘든 작업이지만, 직접 마

코프 모델으로 표현하는 것보다는 훨씬 노력이 절감될 수 있다. <그림 2-3>은 이렇게 작성된 고장 수목을 간략화하여 나타낸 것이다.

이러한 방법은 기존의 고장 수목을 이용한 신뢰도 분석의 틀을 그대로 이용하므로써 분석자들이 이해하기 쉽고, 비교적 정확한 모델링이 가능하다는 장점이 있다. 그러나 소프트웨어의 고장율을 추정하여야 한다는 점이 여전히 문제로 남아 있다.



<그림 2-3> 예제: ACIS에 대한 간략화된 고장 수목

#### 4. BBN을 이용한 방법

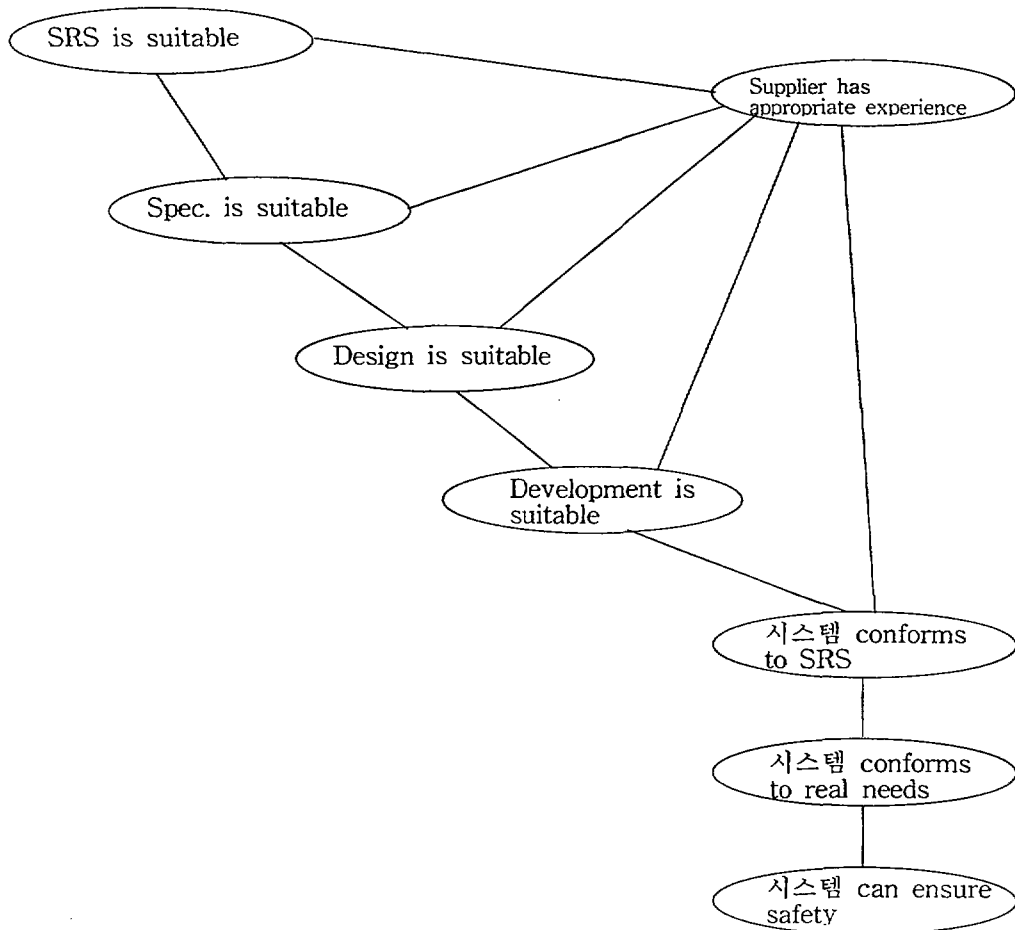
Bouissou 등의 방법은 Bayesian belief network (BBN)을 이용하여 소프트웨어를 포함한 디지털 시스템을 평가하려는 시도이다 [16]. 기존의 방법이 소프트웨어에 내재되어 있는 설계 오류를 평가하기 어려운데 비해서 이 방법은 전문가들의 의견을 효율적·정량적으로 반영할 수 있다는 장점이 있다. 즉, 계통 요구 명세(system requirement specification; SRS)의 적절성, 개발 과정의 적절성, 공급자의 경험, 개발된 시스템이 명세에 부합하는지 여부 등 다양한 인자를 전문가의 의견을 수집하여 평가하는 것이다. 그러나 BBN으로 표현되는 지식들을 정량화하는 과정에서 일관성과 타당성을 확보하는 것이 어렵기 때문에 이 부분을 보완하여야 할 것으로 판단된다.

BBN이란, 일련의 이산 변수(discrete variable)들간의 조건부 확률들을 뜻한다. 이들이 서로 영향을 미치므로 이 변수들을 노드로 하여 그래프를 그려보면 마치 network처럼 보이기 때문에 Bayesian belief network이라고 부르는 것이다. 이러한 서로간의 '관계'에 의한 수학적 확률계산에 관한 법칙은 이미 오래 전에 발견된 것이나, 확률 변수들이 많아 질 경우 그 계산이 기하급수적으로 증가하기 때문에, 현실적인 문제에는 적용되지 못하고 있었다. 그러나 최근 많은 알고리즘들이 개발되고, 전산 기기들의 성능이 향상되면서 다시금 주목을 받게 되었다. 현재에는 수백개 정도의 노드(확률변수)를 가지는 BBN에 대한 계산을 수행하는 정도는 거의 실시간으로 가능한 것으로 알려져 있다. 이러한 연산이 가능해지므로써 실측치 자료들을 이용하여 BBN을 교정하는 본격적인 활용이 이루어질 수 있게 된 것이다.

<그림 2-4>는 프랑스의 EDF에서 계측제어 관련 기기를 선정하기 위해 진행하는 과정을 BBN을 이용하여 도식화한 것들 중 최상위의 개념도이다. 이 개념도는 다시 하위 도식(sub-graph)들로 상세화되는데, 노드들을 연결한 실선은 각 하위 도식들 간에 공유가 존재한다는 의미이다. 즉, 상위 노드들이 서로 공유하는 하위노드가 있다는 뜻이다. <그림 2-4>는 계통 요구 명세의 작성이 적절한지 여부에 대한 노드에서 시작하여 설계의 적절성, 개발 과정의 적절성, 개발된 제품이 명세를 만족하는지에 대한 검증, 실제 필요성에 적합한지의 검토, 대상 시스템에 대한 안전성을 확신할 수 있는지 여부의 전 공정을 그래프의 노드로 나

타내고 있다. 이러한 다양한 활동들을 노드를 통해서 표현하고, 각 노드 내부에서는 다음 노드로 연결되는 하위노드가 있어, 최종 목적이 되는 노드로 필요한 정보를 전달하는 구조를 가지고 있는 것이다. 상세화된 하위노드의 구성은 본 보고서에서는 소개하지 않는다.

BBN을 이용할 경우, 일반적으로 정량화하기 힘든 부분들을 확률이론과 전문가의 힘을 빌어 정량화할 수 있다는 장점이 있다. 그러나 BBN으로 표현되는 지식들을 정량화하는 과정에서 일관성과 타당성을 확보하는 것은 별개의 문제가 된다. 즉, 개념도를 그리고 BBN 그래프를 작성한 후, 실제로 각 이산 변수들의 조건부 확률을 도출하는 과정이 정형화되어 객관성을 확보할 수 있어야 할 것으로 판단된다.



<그림 2-4> 예제: EDF의 계측제어 기기 선정 과정에 대한 BBN 최상위 모델

## 제 5 절 결정론적 방법과 확률론적 방법

주로 소프트웨어에서 발생하는 문제이다. 소프트웨어의 특징인 마모(wear-out)가 없다는 점을 고려하게 되면, 소프트웨어를 결정론적인 것으로 취급할 수 있다. 즉, 문제가 있는 소프트웨어라면 항상 고장을 일으키게 되며, 이것이 '일정한 확률로 문제가 될 수도 있고 그렇지 않을 수도 있다'는 확률적인 처리를 따르지 않는다는 것이다. 실제로 영국의 원전 운전 경험을 분석한 결과에 따르면 소프트웨어 오류중 63% 정도가 명세 오류(specification error)이거나 소프트웨어 설계 오류(design error)인 것으로 밝혀졌다 [6]. 이러한 종류의 오류가 많다는 것은 전체 소프트웨어의 고장 양태가 결정론적인 성향을 보이는 이유를 설명해 준다.

그러나, 실제 시스템들은 소프트웨어만을 포함하고 있는 것이 아니며, 하드웨어와 함께 시스템을 구성하고 있다. 그러므로 이 전체를 확률적인 프로세스로 표현이 가능하다는 연구 결과도 있다 [23].

# of test cases =  $[\log(1-\text{confidence level}) / \log(1-\text{desired failure rate})]$ 의 공식을 이용하면, 99%의 확신도로  $10^6$ 정도의 고장율을 얻기 위해서는 약  $4.6 \times 10^6$ 가지의 테스트 케이스를 생성하여 시험을 한 결과가 전혀 고장이 없는 것으로 나와야 한다 [24]. 이렇게 하기 위해서는 1분에 하나씩의 테스트 케이스를 생성하고 시험을 수행한다고 가정하더라도 9년의 시간이 걸리게 된다. 그러므로 소프트웨어에 대해서 전적으로 테스트에만 의존하여 최종 결과물의 고장율을 추정하는 것은 비현실적인 작업이다.

최근의 연구동향은 이러한 두가지 접근 방법을 모두 수용하는 쪽으로 기울고 있다. 즉, 소프트웨어에 대해서 하드웨어와 동일한 방법으로 확률적인 분석만을 수행하는 것은 곤란하다는 점에 대부분 동의하여, 확률적 분석 방법과 동시에 전 생명주기(life cycle)에서 고장율에 영향을 줄 수 있는 요소들을 분석·고려하는 방법론이 제시되고 있다. 최근에 발간된 대표적인 표준들인 ISA의 SP84 (1996)와 IEC의 61508 (1998)에 이러한 경향이 반영되어 있다 [17], [25].

근래의 연구동향은 대체로 하드웨어와 소프트웨어를 포함한 전체 시스템의 구조와 기능을 정리하고 그 고장 유형(failure mode)에 대한 분석을 수행한 후, 신뢰도 모델링을 수행하는 방향으로 정리되고 있다. 이때 하드웨어에 대해서는 무

작위성 고장에 의한 고장율을 고려하여 주어야 하며, 소프트웨어에 대해서는 설계 오류와 코딩 오류에 의한 고장을 주로 고려하게 된다. 이렇게 시스템의 여러 가지 측면을 생명 주기까지 고려하여 분석하고자 하는 동향을 단계별 접근법(Layered approach)라고 하는데, Karydas와 Brombacher가 정리한 단계별 접근법의 적용 순서와 개요는 다음과 같이 나타낼 수 있다 [17].

1) 시스템 구조 모델링(architecture modeling):

Block diagram 등을 이용하여 개략적인 모델을 만든다.

2) 하드웨어 고장 유형 및 확률(modes & probabilities)분석:

무작위성 고장에 의한 고장율을 추정한다.

이때 공통원인 고장(CCF)과 진단기능에 의한 회복(diagnostic coverage)까지 고려해 주어야 한다.

3) 시스템 전체의 고장 유형을 분석(systemic failure mode analysis):

인간 운전원의 오류(human error)와 소프트웨어의 오류를 방지하기 위한 노력의 정도 등을 평가하여 반영한다.

4) 신뢰도 모델링(reliability modeling) 수행:

기존의 시뮬레이션 방법론이나 신뢰도 블록도(RBD), 고장 수목 분석(FTA), 마코프 모델 등을 이용한다.

## 제 3 장    디지털 계측제어 기기의 고장내구성 기능에 대한 PSA 방법론

원자력 발전소의 경우와 같이 안전성이 중요시되는 시스템에서는 안전 관련 계통을 다중으로 처리하게 된다. 안전 관련 디지털 계측 제어 시스템의 경우에는 전체 시스템에서 다중성을 가질 뿐만 아니라 내부의 소프트웨어나 하드웨어에서도 다중성을 가지거나 자기 감시 등 여러 가지 고장내구성 기법들이 적용된다. 이러한 기법들이 적절히 평가될 경우, fail-safe 개념에서 큰 신뢰도 향상을 기대할 수 있다.

전체 시스템의 신뢰도 분석에 있어서 이러한 다중·자기감시 기능의 신뢰도는 대단히 큰 영향을 미친다. 물론 자기감시 기능 자체에 오류가 있는 경우는 오히려 그 신뢰도를 크게 저하시킬 가능성도 있다. 그러나 일반적으로 디지털 계측제어 시스템에 고장내구성 기능을 추가함으로써 인해 시스템의 복잡도가 높아지더라도, 그것은 자기감시로 인한 신뢰도 향상의 장점을 상쇄할 만한 것은 아니기 때문에, 이러한 기법의 적극적인 도입이 강조되고 있다. 그러기 위해서는 먼저 정확한 평가 방법론의 개발이 선행되어야 한다.

고장내구성 기법을 적용할 경우, 시스템의 구조가 복잡하게 되고, 내부의 논리 흐름이 불연속적이 되는 경우가 많아 확률적 정량 평가에 난점으로 작용하게 된다. 소프트웨어로 구현된 고장내구성 기법들은 매우 다양한 알고리즘을 적용할 수 있기 때문에 분석의 난이도가 더욱 높아진다. 그러므로 이러한 기법들에 대한 체계적인 분석 및 PSA 적용성 검토가 필수적이다.

고장내구성 기법의 유효범위(coverage) 추정치에 의한 시스템의 신뢰도의 변동이 크므로 이에 대한 검토가 필요하다. 유효범위를 추정하기 위해서는 추상적인 고장내구성 기법을 구체적으로 모델링하는 것이 필수적이다. 이렇게 적절한 신뢰도 분석 모델을 도입하고 그에 상응하는 유효범위를 적절히 추정하는 것이 디지털 계측제어 시스템의 신뢰도 분석에 대단히 중요한 역할을 한다. 즉, 추정이 가능한 유효범위 및 확률 모수들만으로 시스템을 모델링할 수 있도록 적절한 분석 모델을 선정하여야 한다.

3.2절과 3.3절에서는 각각 하드웨어 및 소프트웨어에 의한 고장내구성 기능의 구현과 그에 대한 분석 방법을 다루었고, 3.4절에서는 시스템 차원에서의 고장내구성 기능에 대해 다루었다.



## 제 1 절 개요

고장 내구성이 있다는 개념은 결함의 존재 가능성을 인정하고 그에 대해 보다 강인한 특성을 갖는 것을 말한다. 결함에 대해 강인한 특성을 갖는 방법을 두 가지로 분류해 볼 수 있는데, 첫번째는 소재나 알고리즘의 특성상 어느 정도 수준의 외란(disturbance)에는 근본적으로 영향을 받지 않는 경우이고, 두번째는 능동적으로 결함의 발생여부를 감시하고 대처할 수 있는 기능을 갖춘 경우이다.

능동적인 고장 감시 기법들을 다시 두가지로 분류하여 파악할 수 있는데, 회로 수준(circuit-level)의 기법과 시스템 수준(system-level)의 기법으로 구분할 수 있다. Error detecting codes for memories, parity bits for data buses, self-checking circuits 등이 회로 수준의 기법들의 예이며, Capability-based addressing, watchdog timers, fault-tolerant data structures, use of replication (N-version programming 등)이 시스템 수준의 기법들의 예이다. 본 보고서에서 다루는 부분은 시스템 수준의 기법들이다. 회로 수준의 기법들은 하드웨어의 신뢰도 계산에서 반영된 것으로 보기 때문이다.

디지털 계측제어 분야에서 고장내구성 시스템을 설명하면서 watchdog이라는 용어가 많이 사용된다. 그러므로 먼저 watchdog이라는 용어를 정의할 필요가 있는데, 일반적인 고장내구성 시스템에서 능동적으로 그 기능을 구현하는 부분이 바로 watchdog이다. Watchdog은 결함에 대한 감시 및 대처의 역할을 수행하도록 설계된다. 감시 및 고장에 대한 대처의 방법은 적용의 대상에 따라서 다양하게 구현될 수 있다.

Programmable logic controller (PLC) 등의 경우에는 watchdog은 시간 초과(time-over)를 감시하여 시스템 정지(halt)를 방지하는 루틴을 일컫는다. 이 경우에는 대개 감시 타이머(watchdog timer)라고 부른다 [26]. 소프트웨어 분야에서는 출력값을 감시하여 기준치를 초과할 시에는 다른 알고리즘과 프로그램 언어로 짜여진 모듈로 넘기는 역할을 담당하는 모듈을 watchdog이라고 한다 [27]. 이러한 복구 블록(recovery block)을 따로 두는 방법이외에도 여러 가지 버전의 소프트웨어를 설치하는 N-version 소프트웨어 기법 등의 방법으로 고장내구성 기능을 갖추기도 한다. 또 대기 시스템을 가지는 경우에는 시스템 정상 동작 여

부를 감시하면서 비정상 동작으로 판단될 때에는 대기 시스템으로 기능을 넘기는 역할을 하는 시스템을 watchdog이라고 부른다. 다중 시스템의 경우에는 watchdog의 역할을 하는 시스템을 따로 두기보다는 대기 시스템이 감시 신호(heart bit 등)를 이용하여 watchdog의 기능을 수행하는 형태가 많다.

이상을 종합하면, watchdog이란 용어는 시스템이 정상적인 기능을 수행하는지를 확인하기 위해서 하드웨어와 소프트웨어 상에 구현된 알고리즘의 총칭으로 사용되고 있으며, 하드웨어, 소프트웨어, 시스템 차원에서 두루 활용되고 있다. 디지털 시스템이 가지는 고유한 장점인 '자기 검사(self-testing) 기능'을 수행하는 부분으로 해석할 수 있다. 기존의 아날로그 시스템에서는 surveillance test를 포함한 주기적인 검사만이 가능하였다. 그러나 디지털 시스템의 경우에는 이러한 자기 검사 기능이 추가됨으로써, 연속적인 검사(continuous testing)를 구현할 수 있다.

이 watchdog은 원전과 같이 다중 시스템을 가지는 경우에 그 신뢰도 계산에서 반드시 고려되어야 한다. 전체 시스템에서 다중성을 가질 뿐만 아니라 내부의 소프트웨어나 하드웨어에서도 자기검사 기능을 가지므로 적절히 평가될 경우, fail-safe 개념에서 큰 신뢰도 향상을 기대할 수 있다. 이러한 자기검사 기능을 가진 디지털 시스템을 안전 관련 기기에 이용하는 것에 대한 미국의 NRC의 입장은 다음과 같다 [26].

*"The positive aspects of self-test features are not compromised by the additional complexity that may be added to the safety system by the self-test features."*

디지털 계측제어 시스템에 자기검사 기능을 추가함으로써 인해 시스템의 복잡도가 높아지더라도 그것은 자기검사로 인한 신뢰도 향상의 장점을 상쇄할 만한 것이 아니라는 입장이다. 즉, 이러한 자기검사 기능의 적극적인 도입을 권장하고 있는 것이다.

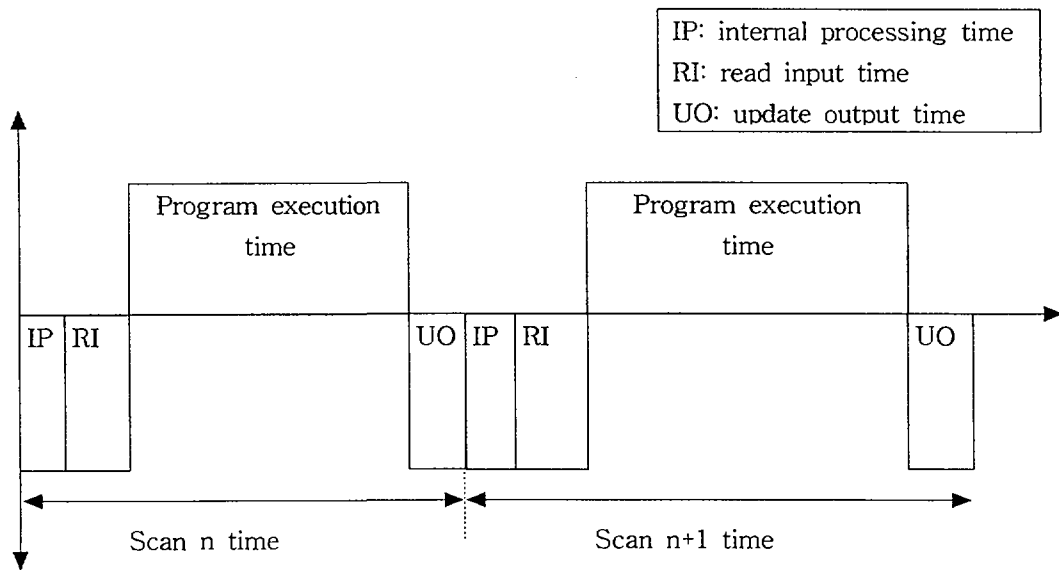
그러나 만약 이러한 자기검사 기능 자체나 복구 블럭에 오류가 있는 경우는 오히려 그 신뢰도를 크게 저하시킬 가능성이 있으므로 PSA 수행 시에 적절한 처리가 필수적이라 하겠다. 이 문제에 대해 미국의 NRC의 경우, 자기검사 시스템(또는 소자)에 대해 감시대상이 되는 시스템과 같은 수준의 안전등급 시스템을 적용해야 한다고 밝히고 있다.

하드웨어, 소프트웨어, 시스템 각각에 대하여 고장내구성 기능이 있는 경우의

처리에 대한 연구([27], [28], [29], [30] 등)는 다수가 수행된 바 있는데, 전체를 포괄할 수 있는 일반적인 틀(framework)은 아직 제안되지 않고 있다. 본 연구를 통해 개발하여야 할 부분이다.

## 제 2 절 감시 타이머 및 자기검사 기법들

일반적으로 산업 현장에서 많이 이용되는 대표적인 컴퓨터 기반 시스템 (computer-based system)인 PLC의 경우에는 감시 타이머라는 장치를 내장하고 있다. 시스템에 이상이 생겼을 경우, 이를 감지해 내기 위한 가장 기초적인 장치이다. 감시 타이머의 적용 범위는 PLC에 국한되는 것이 아니라 전반적인 컴퓨터 시스템 모두에 해당한다. 그러나 PLC와 같이 주기적인 반복실행(cyclic operation)을 위주로 하는 시스템에 특히 효과적으로 적용될 수 있다. 감시 타이머는 time set-point가 고정된 것과 가변적인 것의 두가지 종류로 구분할 수 있다 [31]. 최근의 상용 PLC들은 대개 이 두가지 감시 타이머를 모두 포함하고 있다.



<그림 3-1> PLC의 작업 수행 시간 개념도

고정 감시 타이머는 하드웨어 interrupt를 발생시키는 일종의 interval timer로서, time-over를 감시하여 시스템 정지를 방지하는 작용을 한다. 대개의 경우 이 고정 감시 타이머의 시간 설정치(time set-point)는 제조과정에서 고정되어 생산되므로 사용자가 임의로 변경하는 것은 불가능하다. <그림 3-1>은 일반적인 PLC의 작업 수행 과정을 도시한 것이다.

내부적인 처리 시간, 입력을 읽어 들이는 시간, 실행 코드를 수행하는 시간, 출력값을 갱신하는 시간을 합쳐서 하나의 scan time이라고 하는데, 고정 감시 타이머에서는 이 scan time이 미리 정해진 시간 설정치 보다 길어지게 되면 PLC를 정지시키고 고장 신호를 출력하게 된다. 즉, 하나의 scan내에 있는 네가지 요소중에 어떤 부분에 문제가 있어서 시간을 초과하게 되더라도 PLC 전체를 정지시키는 것이다. 이 고정 감시 타이머는 CPU가 정지된다고 하더라도 작동하게 된다.

소프트웨어를 이용하여 감시 타이머의 시간 설정치를 조정할 수 있는 경우를 가변 감시 타이머(software-based watchdog timer)라고 하는데, 이것은 scan time이 지정된 이상으로 길어지는 것을 방지하기 위한 목적으로 사용된다. 무한 루프를 돌거나 하는 것을 미연에 방지하기 위한 것이다. 적용 방법을 개략적으로 설명하면 다음과 같다. 시스템의 응용프로그램이 실행되기 전에 미리 내장 counter를 특정한 값으로 설정한다. 그리고 응용프로그램의 작동이 시작되면 내부의 clock에 의해 내장 counter의 값이 일정시간 마다 1씩 감소하게 된다. 이 값이 0이 되기 전에 응용프로그램이 종료하여 내장 counter의 값을 처음의 값으로 reset해 주지 않으면 counter의 값이 음수로 내려가게 되는데, 이를 검출하여 PLC를 정지시키고 경고 신호를 출력하게 된다.

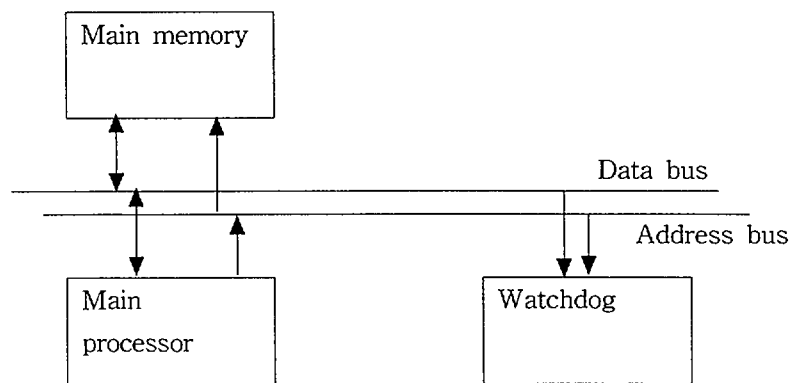
위의 두가지 방법 이외에도 PLC 외부에 second line of defense의 성격을 가진 감시 타이머를 설치하기도 한다.

이러한 감시 타이머들의 사용은 “정해진 시간을 초과하는 코드 실행은 시스템의 오류를 의미한다”는 가정하에 이루어진다. 그러나 모든 시스템의 오류가 시간 지연을 유발하는 것은 아니므로, 감시 타이머의 유효범위는 제한되어 있다. 감시 타이머의 출력(정상/비정상 판단) 신호는 다양한 형태로 제공될 수 있다. 시스템을 reset시켜 재시도를 하도록 하는 것이 일반적인 용도이다. 일시적인 하드웨어의 문제 때문에 시간 설정치를 넘긴 것이라면 이러한 재시도를 통해 원상복구될 수 있는 것이다. 따라서 PLC와 같이 주기적인 반복실행을 수행하는 경우에는 우

발적 하드웨어 오류에 대한 효과적인 대처 방안일 수 있는 것이다. 응용 대상에 따라 감시 타이머의 출력 신호가 시스템을 reset시키는 것이 아니라 stop시키고 고장 신호를 외부로 발생하도록 할 수도 있다. 원자력 발전소의 보호 시스템과 같은 안전관련 시스템에 적용할 때에는, 감시 타이머의 출력 신호가 원자로 정지 신호와 연동되도록 설치할 수도 있다. 만약 여분의 대기 시스템(다중 시스템)이 있는 경우라면, 대기 시스템이 주 시스템의 watchdog 신호를 감시하도록 하기도 한다.

어느 응용 사례의 경우이나 감시 타이머는 단일 시스템의 신뢰성 관점에서는 도움이 되지 못한다. 오류를 방지하는 것이 아니라 오류 발생후에 감지해 내는 역할을 수행하기 때문이다. 그러나 시스템의 오류를 조기에 인지하고 대처할 수 있으므로 전체 시스템 차원에서는 신뢰성을 높이는 효과와 수리 시간을 줄여 가용도를 향상시키는 효과가 있다. 응용 사례에 따라서는 (원자력 발전소의 보호 시스템 등 dangerous failure만이 관심 대상이 되는 경우) 감시 타이머의 유효범위에 해당하는 만큼의 시스템 신뢰도 향상을 가져오기도 한다. 따라서 감시 타이머가 포함된 시스템의 신뢰도 분석은 이러한 '고장 감지후의 대처 방법'을 고려하여 수행되어야 하며, 고장 검출 유효범위를 규명하는데 초점을 맞추어야 한다.

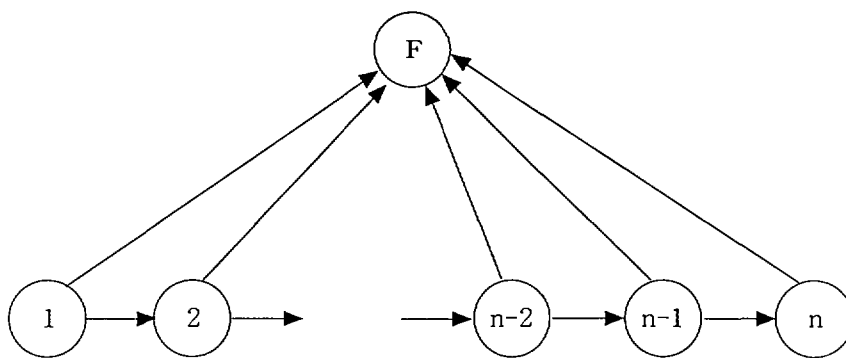
한편, 시스템 내부에 감시 프로세서(watchdog processor)를 두기도 한다. 감시 프로세서는 기존의 주프로세서(main processor) 외에 별도로 간단한 프로세서를 설치하여 고장을 감시하는 장치이다. <그림 3-2>는 이러한 감시 프로세서의 설치 형태를 도시한 것이다.



<그림 3-2> 감시 프로세서

이러한 감시 프로세서는 초기의 설정시에 감시해야 할 정보를 입력해 주고, 실제 시스템 운용에 들어가면 온라인 상태에서 연속적으로 시스템의 감시를 수행하는 역할을 하게 된다. 이것은 전술한 감시 타이머와 완전 다중 사이의 중간 형태라고 볼 수 있다. 즉, 완전 다중 시스템을 구현하는 비용을 절감하기 위해 간단한 감시 프로세서를 설치하는 것이다. 비교적 저렴한 비용으로 단순한 감시 타이머보다는 넓은 범위의 결함을 감시할 수 있다는 장점이 있다. 감시 프로세서를 별도로 뒀으로써 공통원인 고장의 확률을 낮출 수 있다는 장점도 있다 [32]. 이 감시 프로세서가 감시를 수행하는 알고리즘은 여러 가지가 있다. Memory access를 감시하거나, control flow나 control signal을 감시하기도 한다. 또는 주 프로세서에 의해 계산된 결과값의 적절성(reasonableness of results)을 감시하기도 한다.

감시 프로세서를 다중으로 장치할 수도 있다. 감시 프로세서가 고장 상태일 때 다른 감시 프로세서가 작동할 수 있도록 여러개를 예비로 장치하는 것이다. Imaizumi 등은 이렇게  $n$ 개의 감시 프로세서를 가지는 시스템의 신뢰도를 제안하였는데, 주프로세서가 memory access나 control에 실패하는 실패율은 일반적인 확률분포  $F(t)$ 를 따르고, 감시 프로세서의 고장률은 지수분포를 따른다는 가정하에 접근하였다. <그림 3-3>은 이러한 시스템의 transition diagram을 도시한 것이다.



<그림 3-3>  $n$ 개의 감시 프로세서를 가진 시스템에 대한 모델

<그림 3-3>에서 각 state  $i$  는 각  $i$ 번째 감시 프로세서가 주프로세서를 감시하고 있는 상태를 나타낸다. F는 주프로세서가 고장이 나서 작동불능이 된 상태를 나타낸다. 이 모델에서  $n=1$ 로 하면 일반적인 단일 감시 프로세서의 경우가 된다. 이 모델에서는 각 감시 프로세서가 고장을 검출하는데 성공할 경우 즉각 시스템을 reset시키며, 이때 걸리는 시간은 극히 짧아서 무시할 수 있는 것으로 가정하였다. 원자력 분야의 안전관련 시스템에서는 다른 가정이 필요할 것으로 생각되지만, 참고를 위해 Imaizumi의 결과를 소개하면 다음과 같다. 각 state간 천이시간 분포를 수식으로 표현하면 다음과 같다. 자세한 유도과정은 생략하기로 한다.

$$q_{ii}(s) = pf(s+a)$$

$$q_{i,F}(s) = \frac{1}{1-pf(s+a)} [(1-p)f(s+a) + (1-\theta)(f(s)-f(s+a)) + \frac{\alpha\theta}{\alpha-\beta}(f(s+\beta)-f(s+a))]$$

$$q_{i,i+1}(s) = \frac{1}{1-pf(s+a)} \frac{\alpha\beta\theta}{\alpha-\beta} \left[ \frac{1-f(s+\beta)}{s+\beta} - \frac{1-f(s+a)}{s+a} \right]$$

$$q_{n,F}(s) = \frac{1}{1-pf(s+a)} (f(s)-pf(s+a))$$

$q(s)$ : state간 천이시간 분포  $Q(t)$ 의 Laplace 변환

$f(s)$ :  $F(t)$ 의 Laplace 변환

$p$  : 주프로세서의 고장을 감시 프로세서가 감지할 확률 (coverage)

$\theta$  : 감시 프로세서가 자신의 고장을 감지할 확률

$\alpha$  : 지수분포로 가정된 감시 프로세서의 고장을

$\beta$  : 선행 감시 프로세서의 고장을 감지하고 차순의 감시 프로세서가 주프로세서를 reset시킨 후 자신이 주프로세서의 고장을 감시하는 일련의 절차를 처리하는데 걸리는 시간 분포를 지수 분포로 가정 ( $1-e^{-\beta t}$ )

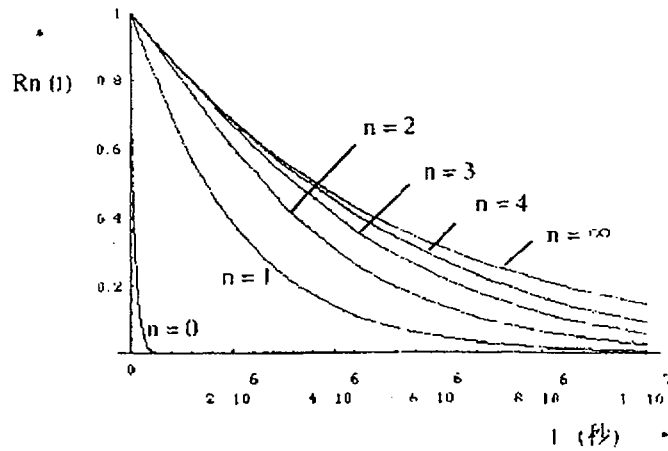
예를 들어서,  $F(t)$ 가 지수분포( $1-e^{-\lambda t}$ )라는 가정을 도입하면 다음의 식에 의해 1개의 주프로세서와  $n$ 개의 감시 프로세서를 가지는 시스템의 고장 시간 분포  $H(t)$ 는 state간의 천이시간 분포  $Q(t)$ 의 조합으로 구할 수 있게 된다. 이때  $h(s)$

는  $H(t)$ 의 Laplace변환이다.

$$H(t) = Q_{1,F}(t) + Q_{1,2}(t) Q_{2,F}(t) + \dots + Q_{1,2}(t) Q_{2,3}(t) \dots Q_{n-1,n}(t) Q_{n,F}(t)$$

$$h_n(s) = \frac{\lambda}{s+\lambda} \left\{ 1 - \frac{sp}{s+\alpha+\lambda(1-p)} \times \sum_{j=0}^{n-1} \left[ \frac{\alpha\beta\theta}{(s+\alpha+\lambda(1-p))(s+\beta+\lambda)} \right]^j \right\}$$

[28]에서  $\beta$ 는 시스템 고유 클럭의 100배 정도를 설정하는 것이 적절하다고 밝히고 있으며,  $\alpha$ 와  $\lambda$ 는 감시 프로세서와 주프로세서의 평균 고장 시간을 이용하여 구하여야 한다. 수치적인 모사의 결과를 나타낸 그래프를 <그림 3-4>에 옮겨 놓았다.



<그림 3-4> n개의 감시 프로세서를 가진 시스템의 신뢰도  
( $p=0.99$ ,  $\theta=0.8$ ,  $1/\lambda=1$ 일,  $1/\alpha=30$ 일,  $1/\beta=1/300,000$ 초)

한편, 결함 검출 이후의 회복(recovery)까지 고려한 [27]의 실험중 하드웨어(disk)의 결함에 관련한 부분에서는 결함의 검출확률(유효범위)의 변화에 따라 ( $0 < p < 1$ ) 하드웨어의 MTTF가 수배에서 수십배까지 향상되는 것으로 나타났다.

### 제 3 절 고장내구성 소프트웨어

디지털 시스템의 신뢰도 향상을 위해 소프트웨어에 적용될 수 있는 여러 단계



에서의 기법들을 하드웨어에 적용 가능한 기법들과 대비하여 정리하면 <표 3-1>과 같다 [33]. Lyu 등은 결함에 대한 대응을 4단계로 나누어 생각하였는데, 첫번째 단계는 결함을 회피하도록 설계하는 방법이다. 소프트웨어의 경우 철저한 품질관리(lifecycle control)와 확인·검증(verification and validation), 모듈화를 통한 설계 단순화 등이 이에 해당한다. 두번째 단계는 결함을 검출할 수 있도록 하는 것인데, 소프트웨어의 실행 결과를 감시하거나 시간지연을 감시하는 방법이 이에 해당한다. 세번째와 네번째 단계는 결함의 영향을 제거하기 위한 조치에 해당한다. 이러한 조치들을 크게 두가지로 분류하여 masking redundancy와 dynamic redundancy로 명명하였는데, 전자는 다중 정보의 가중치를 조절하여 결함이 있는 정보(faulty information)의 영향을 극소화하는 방법이고, 후자는 적극적으로 결함의 원인을 찾아 해결하거나 재시도하는 방법이다. 소프트웨어의 경우 두가지 방법 모두 하드웨어의 경우에 비해 상대적으로 구현이 용이하다.

<표 3-1> 디지털 시스템의 하드웨어와 소프트웨어에서의 신뢰성 향상 기법 [33]

분류	하드웨어	소프트웨어
Fault avoidance	Quality changes Component integration level	Software engineering -modularity
Fault detection	Duplication Error detection codes Self-checking and fail-safe logic Watchdog timers and timeouts Consistency and capability checks Processor monitoring	Program monitoring Watchdog timers and timeouts
Masking redundancy	Error correcting codes Masking logic	Algorithm construction
Dynamic redundancy	Reconfigurable duplication Backup sparing Graceful degradation Reconfiguration Recovery	Forward error recovery Backward recovery -retry -checkpointing -journaling -recovery blocks

소프트웨어 분야의 고장 내구성 구현 방법에는 대표적으로 복구 블럭을 두는 경우와 N개 버전의 소프트웨어를 다중으로 두는 방법이 있다. 복구 블럭 방법은 전형적인 backward recovery 방법인데, acceptance test를 통해 결함 발생을 감지하면, 결함이 발생하기 전의 상태로 시스템을 되돌려 다시 실행을 시킨다. 이때 다중성이 없는 경우에는 같은 코드를 반복 실행하게 되는데, 이것은 소프트웨어 자체의 문제 해결보다는 하드웨어에서 발생할 수 있는 무작위성 결함에 대응하기 위한 목적이 더 크다. 일반적인 경우에는 acceptance test에서 결함이 발견되면, 다중 모듈을 실행시킨다. 즉, 출력값을 감시하여 기준치를 초과할 시에는 다른 알고리즘과 프로그램 언어로 짜여진 모듈로 넘기게 된다 [27], [34].

동일한 목적을 가진 여러 가지 버전의 소프트웨어 코드를 설치하여 다중성을 주는 방법을 N-version 소프트웨어 기법이라고 한다 [35]. 이것은 Forward error recovery의 전형적인 경우이다. N개의 코드를 동시에 실행시킨 후, 최종단에서 voting을 하여 결과값을 출력하게 되므로 결함이 있는 코드의 실행 결과는 최종 결과에 반영되지 못하게 된다(masking).

위에 설명한 여러 가지 고장내구성 기법들을 포함하는 소프트웨어의 경우, 그 효과를 모델링하는 것이 간단하지 않다. 제 2장에서 언급한 바와 같이, 고장내구성 기법에 대한 고려가 없는 보통의 소프트웨어의 신뢰도 모델링에 대해서조차 아직 확립된 방법론이 없기 때문에, 이러한 보통의 소프트웨어들을 하나의 모듈로 가지는 고장내구성 소프트웨어를 모델링하는 것은 어려운 작업이다. 현재 발표되고 있는 연구결과들([27], [34], [35], [36], [37])은 공통적으로 확률함수로 표현되는 고장 분포를 이용하고 있는데, 이것은 하드웨어의 확률론적 고장 모델에서 그대로 원용해 온 것이다.

결함 주입 기법을 이용하여 고장내구성 시스템의 고장률을 시험한 연구의 결과가 Hocenski 등에 의해 발표되었는데, 이 시험의 결과중 소프트웨어의 다중성이 있는 경우와 없는 경우만을 비교할 때, 다중성이 있는 경우는 masking effect가 뚜렷하게 나타난 것으로 나타났다. 즉, 다중성이 있는 소프트웨어의 경우에는 그렇지 않은 소프트웨어에 비해 전체 고장 수가 수배정도 감소한 것으로 나타났다 [29].

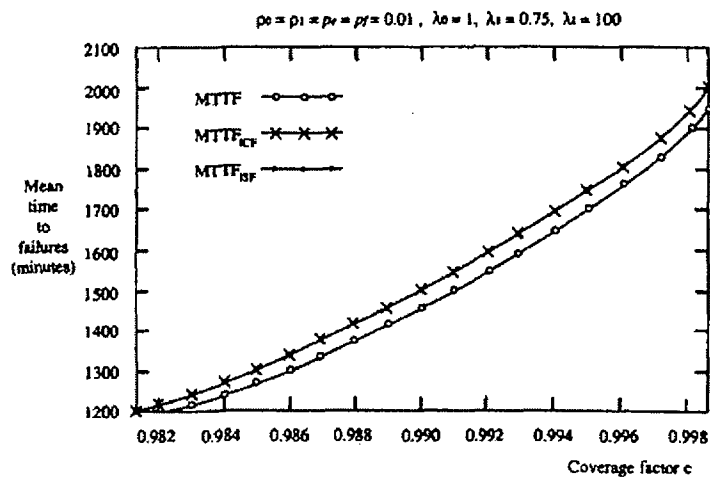
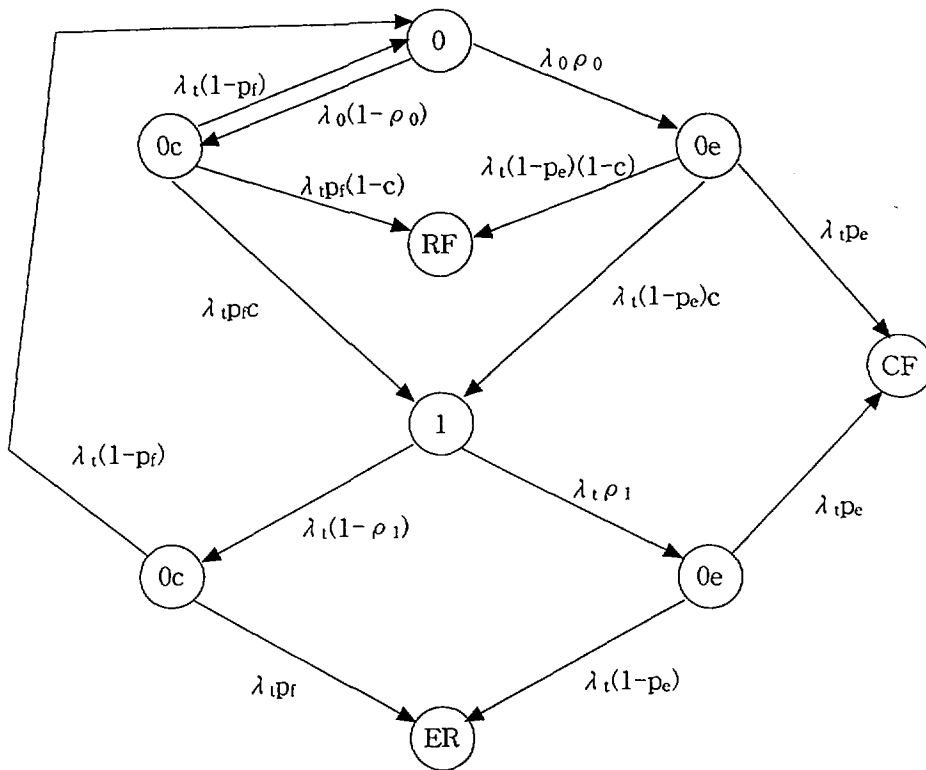
복구인자(recovery factor) 개념을 이용하여 소프트웨어의 MTTF를 시험한 연

구가 Choi 등([27])에 의해 발표되었다. 실험의 대상은 1개의 주모듈과 1개의 다중 모듈을 가지는 복구 블록 소프트웨어이다. 복구 인자( $c$ 값)이 변화함에 따라 소프트웨어의 MTTF가 급격히 변화하는 것을 관찰할 수 있다 (<그림 3-5>). 주모듈과 다중 모듈은 각각 0과 1로 표기되었으며, 침자  $c$ 와  $e$ 는 각각 정확한 결과를 출력한 상태와 잘못된 결과를 출력한 상태를 나타낸다. RF와 CF는 복구가능 고장 상태(recovery failure state)와 치명적 고장상태(catastrophic failure state)를 나타낸다. 각 모듈이 잘못된 결과를 출력할 가능성을  $\rho_0$ 와  $\rho_1$ 로 나타내었다.  $p_r$ 는 acceptance test를 통해 잘못된 출력값으로 판단되는 확률을 나타낸다.  $p_e$ 는 acceptance test가 잘못된 출력값을 검출하지 못할 확률을 나타낸다.

출력값에 오류가 있는 것으로 판단되면 다중 모듈로 전환을 시도하는데, 이 전환이 성공할 확률을  $c$ 로 표기하였다. 이때, 각 모듈의 수행시간( $1/\lambda_0$ ,  $1/\lambda_1$ )과 acceptance test에 걸리는 시간( $1/\lambda_c$ )이 지수분포를 따른다는 가정을 하였다. 모델링에 대한 상세한 설명은 생략한다.

마코프 모델은 비교적 다양한 대상에 대해서 정확한 모델링이 가능하다는 장점이 있지만, 여러 가지 상황을 고려하다 보면 그 상태공간이 지나치게 커져서 쉽게 다룰 수 없게 될 수 있으며, 시스템 내의 각 component의 신뢰도 변화나 서로간의 상관관계 등을 고려할 수 없다는 단점이 있다.

Gokhale 등은 마코프 모델의 이러한 단점을 극복하기 위해 이산 사건 시뮬레이션 기법을 도입하였다 [36], [37]. 대상 시스템에 대해 제어흐름도(control flow diagram)나 데이터흐름도(data flow diagram)를 작성한 후, non-homogeneous continuous time Markov chain (NHCTMC)을 적용하여 수치적인 모델링을 수행하였다. Gokhale 등은 distributed recovery block (DRB), N-version programming (NVP), N self-checking programming (NSCP)의 3가지 고장내구성 기법에 대해 적용을 한 예제를 제시하였는데, 결과는 대체적으로 NVP, DRB, NSCP의 순으로 오류를 감소시킬 수 있는 것으로 나타났다. 그들은 이 수치 해석에서 각 모듈간의 연계 결함(related fault)의 확률을 정의하여 이용하였는데, 이는 일반적인 복구 인자에 해당하는 것이다. 이 복구 인자가 높을수록 (연계 결함이 적을수록), 고장내구성 기법들이 단일 모듈의 결과보다 신뢰성이 높은 것으로 나타났다.



<그림 3-5> 1개의 다중 모듈을 가지는 복구 블럭 소프트웨어에 대한 마코프 모델과 계산 결과 예시 [27]

복구 블럭과 N-version programming 기법을 혼합하여 사용할 경우에 대한 신뢰도를 수치해석을 통해 접근하려는 시도가 Wu 등에 의해 제시되었다 [34]. 적용 비용을 고려한 분석이기 때문에 실질적인 적용까지는 많은 가정들이 필요하였지만, 대략적인 결과는 N-version programming 기법을 이용하는 것이 비용·효과 면에서 모두 유리한 것으로 나타났다.

## 제 4 절 시스템 다중성

위의 제 2 절과 제 3 절에서 설명한 내용은 하드웨어의 고장 오류에 대한 감시와 소프트웨어의 고장 오류에 대한 감시 기법들과 그 모델링 기법들에 관한 것이었다. 이러한 방법들은 전 시스템에 대한 다중성을 확보하는 비용과 노력을 절감하면서도 고장 내구성을 확보하기 위한 노력의 결과들이다. 본질적으로 다중성이 확보되지 않은 시스템은 고장내구성을 가지지 않는다 [33]. 그러므로 이러한 다중성의 확보를 저렴한 비용으로 소프트웨어 또는 하드웨어 각각의 차원에서 추구하는 것이 위의 제 2 절과 제 3 절에서 설명한 내용이다. 시스템 전체의 다중성을 확보하는 것이 비용은 많이 들지만 가장 확실한 고장내구성 기법임에는 의심의 여지가 없다. 이러한 시스템 차원의 다중성 확보의 전형적인 형태가 병렬(parallel) 배치나 2/3, 2/4 등의 voting (auction) 방법이다. 이런 경우들에 대한 해석 방법은 잘 설정되어 있으므로 본 보고서에서는 설명을 생략한다.

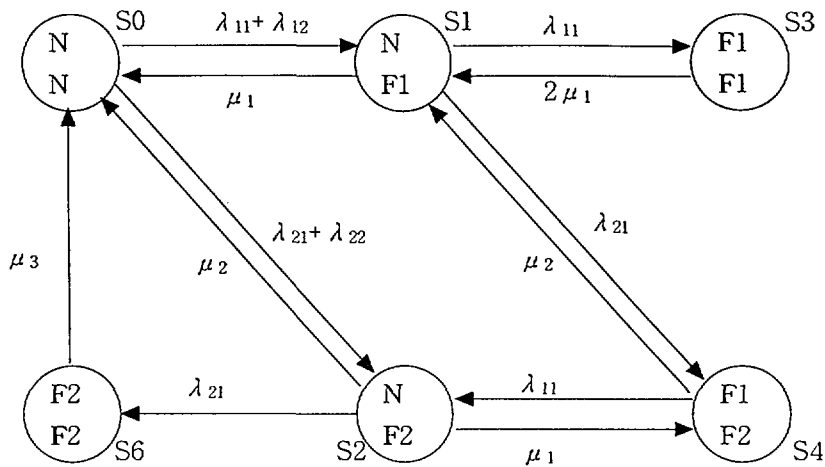
경우에 따라서는 이러한 병렬 배치나 voting 방법을 이용하기 힘든 때가 있는데, 전형적인 경우가 데이터 처리 및 저장(data processing and storage) 시스템이다. 예를 들어 원자력발전소의 계측제어 시스템을 생각해 보자. 원자로 보호 시스템과 같은 once-through 형태의 단순한 신호 처리의 경우라면, 병렬이나 voting의 기법이 유용하게 이용될 수 있다. 그러나 디지털 데이터처리 시스템을 생각해 보면, 각 신호의 처리 때마다 voting을 거쳐 결과를 저장하고, 저장한 결과값을 읽을 때마다 여러 database로부터의 데이터를 voting한다는 것은 현실적으로 불가능하거나 매우 비효율적인 일이다. 이러한 경우에는 hot-standby 백업을 둔다. 즉, 주 시스템(primary system)이 모든 결과를 처리하되, 주 시스템을 감시하는 장치를 두어, 만약 주 시스템의 동작이 비정상적이라고 판단될 때에는, 백업 시스템으로 그 기능을 전환(switch over)하는 것이다. 이런 경우, watchdog의 역할을 하는 시스템을 따로 두기보다는 백업 시스템이 작동 신호(heart-bit

등) 감시 기법을 이용하여 watchdog 기능까지 함께 수행하는 형태가 많다. 즉, 백업 시스템이 고장 감시와 고장시의 처리를 함께 수행하는 것이다.

이러한 시스템을 모델링할 때, 하드웨어와 소프트웨어 각자의 고장내구성 기법은 전혀 사용되지 않고, 주 시스템의 고장을 정확하게 감지하여 백업 시스템으로 전환하는 것이 100%의 성공률을 가진다고 가정을 하게 되면, 대단히 간단한 모델이 된다. 그러나 현실적으로 이런 경우는 없으며, 하드웨어와 소프트웨어, 그리고 시스템의 고장내구성 기법들이 혼합·중복되어 적용되는 경우가 많고 백업 시스템이 고장을 감지하여 전환하는 작업이 100%의 성공률을 가지지도 않는다.

Tan은 2가지 고장의 종류를 가지는 시스템에 꼭 같은 warm-standby 백업 시스템이 있는 경우를 마코프 모델을 이용하여 모델링하였다 [38]. 2가지 고장의 종류중 한가지는 일과성 결함(transient fault)으로, self-reset 기능으로 복구가능한 고장이다. 다른 한가지의 종류는 영구 고장(permanent failure)으로, 이 경우에는 repair가 없이는 다시 정상상태로 돌아올 수 없다. Tan이 고려한 시스템에는 self-reset 기능이 있어서 (하드웨어에 감시 타이머가 있는 경우를 생각할 수 있다), 일과성 결함의 경우에는 일정한 비율로 저절로 회복이 가능한 것으로 모델링하였다.

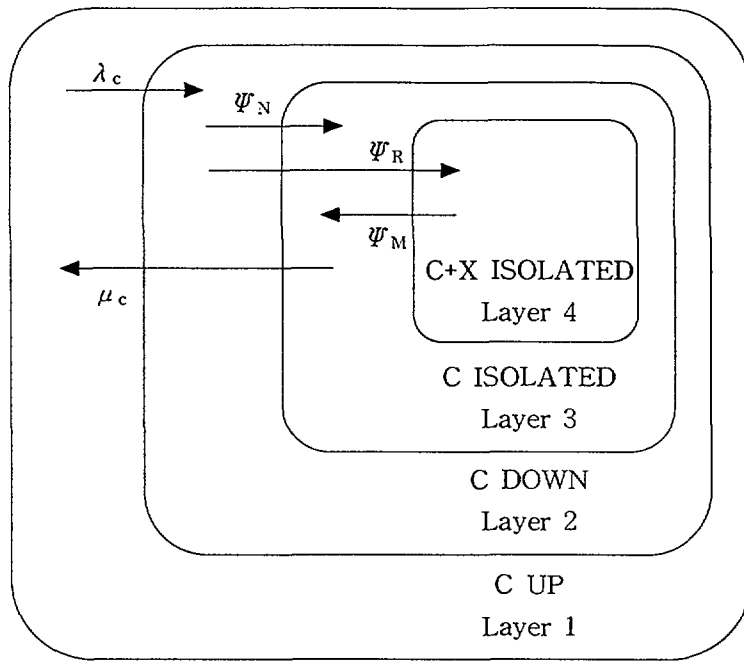
<그림 3-6>에 이와 같은 시스템의 모델링을 도시하였다. 이 모델을 해석하기 위해서는 몇가지 가정이 필요한데, 일과성 결함과 영구 고장의 발생이 확률적으로 독립이라는 가정이 필수적이다. 그리고 Tan은 이상적인 전환 기기(switching device)를 가정하였다. 즉, 고장을 감시하고 백업 시스템으로 전환하는 기기에는 고장이 없고, 전환에 걸리는 시간은 없는 것으로 가정을 하였다. F1은 일과성 결함을 나타내며, F2는 영구 고장의 발생을 나타낸다.  $\lambda_{ij}$ 에서  $i$ 는 고장의 종류가 F1인지, F2인지에 따라 각각 1과 2로 나타내며,  $j$ 는 고장 시스템의 종류가 동작 기기(active unit)에 고장이 나타난 것인지, 대기 기기(standby unit)에 고장이 나타난 것인지에 따라 1과 2로 나타낸다. 한편, 수리율  $\mu_1$ 은 self-reset으로 복구되는 것을,  $\mu_2$ 는 수리를 받아서 복구되는 것을 나타낸다.  $\mu_3$ 는 시스템이 완전히 고장난 후에 수리되는 율을 나타낸다.



<그림 3-6> 2 unit warm-standby 시스템의 마코프 모델 [38]

Tan은 <그림 3-6>의 모델에서 S6만이 고장 state라고 보았으나, 원자력 발전소의 안전 관련 시스템의 경우에는 훨씬 복잡해진다. 즉, 시간 제한(time limit)이 있는 경우에는 일과성 결함이라도 self-reset하는데 걸리는 시간이 길어질 때에는, 시스템이 fail한 것으로 보아야 할 경우가 생길 수 있기 때문이다. 일과성 결함의 복구에 걸리는 시간이 주어진 시간 제한보다 길 때에는, S3과 S4 state도 시스템 고장에 해당될 수 있다. 즉, Tan의 경우와 같이 임의 수리(random repair)를 가정하는 경우에는, 일부분의 일과성 결함은 정상 상태(N)으로 복구되나, 복구에 걸린 시간이 너무 긴 나머지의 일과성 결함들은 정상 상태로 복구되지 못하는 것으로 처리되어야 한다.

한편, Anderson은 일반적인 보호(protective) 시스템의 경우에 대해 상세한 모델링을 제시하였다 [39]. 대상 시스템(또는 component)에 문제가 발생할 경우, 그 시스템을 격리하고 수리하는 과정을 마코프 모델을 이용하여 모델링한 것이다. 하나 이상의 다중 보호 시스템을 포함하며, 공통 모드 고장(CMF)을 고려하였다. 또한 보호 시스템에 대한 점검(inspection)도 포함하여 모델링하였다. <그림 3-7>은 Anderson이 제안한 모델의 개념도이다. 상세한 마코프 모델은 [39]에 제시되어 있다.



<그림 3-7> 다중보호 (redundant protective) 시스템의 모델 [39]

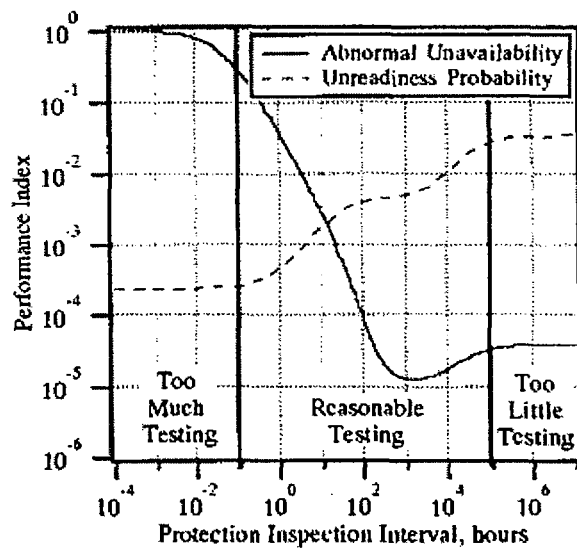
Layer 1은 보호대상 시스템(C)이 정상 가동중인 상태이다. 여기서  $\lambda_c$ 의 비율로 C가 고장난 상태(Layer 2)로 천이한다. 이때 하나 이상의 보호 시스템이 정상적으로 작동하여 C를 격리시키면 Layer 3으로 천이한다. 만약 보호 시스템들이 기능을 원활히 수행하지 못하여 예비 보호 시스템이 작동하여야 할 경우라면, 이 백업 시스템은 C뿐만 아니라 X시스템까지도 함께 격리하게 된다. 이 상태가 Layer 4이다. Layer 2에서 Layer 1으로 바로 복귀할 수는 없으며, 일단 Layer 3나 Layer 4로 격리된 후, 수리를 거쳐 Layer 1으로 복귀하게 된다. 한편, 보호 시스템을 작동 정지 상태로 (unavailable 상태로) 만든 후에야 점검이 가능한 것으로 가정하였으며, 공통의 문제로 인한 보호대상 시스템과 보호 시스템의 고장을 고려하였다. 물론 보호 시스템들 내에서의 공통 모드 고장도 고려하였다.

이상의 과정은 원자력발전소의 노심보호 시스템의 작동 구조와 매우 흡사하다. 원자력발전소에서도 trip signal channel을 격리하여 점검해야 하며, 만약 원자로에 문제가 있을 경우에는 일단 정지시켜 적절한 조치를 취한 후에야 재가동을 시킬 수 있다. Anderson 등은 보호 시스템의 점검 주기를 결정하는 데에 이러한 모델링을 응용하는 예를 보였는데, 고장률, 수리율에 임의의 특정값을 대입



한 예제를 통해 <그림 3-8>의 그래프를 제시하였다. 이것은 2개의 보호 시스템이 존재하는 경우에 대한 것이다. 그림의 실선은 보호 시스템의 불가용도를 나타내며, 점선은 실제 작동이 요구되었을 때 시스템이 작동하지 못할 unreadiness를 나타낸다. 점검 및 시험 주기는 이 두가지를 적절히 고려하여 결정되어야 한다.

한편, <그림 3-8>의 그래프에서 비정상 불가용도의 최저점인  $1.3 \times 10^{-5}$ 은 단일 보호 시스템의 경우( $5.1 \times 10^{-5}$ )에 비해서 4배 가량 낮아진 것인데, 이것은 보호 시스템을 하나 더 추가한 결과이다.



<그림 3-8> 보호 시스템의 점검 주기를 결정하기 위한 Anderson의 예제 [39]

## 제 4 장 단계화 임무 시스템에 대한 분석 방법론

하나의 시스템이 여러 가지의 기능을 수행하여야 할 경우에는 시간을 세분하여 여러 단계로 나눈 후 각각의 단계(phase)에서 한가지씩의 작업을 수행하게 되는데, 이런 시스템을 단계화 임무(phased-mission) 시스템이라고 한다. 이런 경우, 기존의 고장 수목 등의 분석방법으로 상세하고 정확한 신뢰도를 분석하는데는 어려움이 많아 마코프 모델 등 보다 복잡한 모델링 방법을 이용하게 된다.

신뢰도 분석의 상세함의 수준(level of detail)을 높일수록 보다 현실적이고 정확한 분석 결과를 얻게 된다. 단계화 임무 시스템의 경우에는 고장 유형과 그 영향의 범위가 단일 기능 시스템의 경우에 비해 추정하기가 어렵다. 모델링이 개략적으로 수행될 경우, 지나치게 과소 평가되거나 과대 평가된 신뢰도 분석 결과를 얻을 가능성이 높다. 그러므로 정확한 분석 결과를 얻기 위해서는 시스템 내부의 각 세부 기능들의 수준까지 모델링을 수행하여야 한다. 이러한 상세 분석 작업은 다른 시스템에 대한 분석에 비해 지나치게 많은 시간과 비용을 필요로 하게 된다. 그러므로 이러한 모델링을 보다 효율적이고 체계적으로 수행하여 적용성을 높이기 위한 연구들이 수행되고 있다. 본 장에서는 이러한 방법론들을 소개하였다.

### 제 1 절 개요

단계화 임무 시스템이란 한 시스템이 여러 가지 기능(mission)을 가지기 때문에, 각 phase별로 각각의 다른 기능을 수행하는 시스템을 일컫는다. 하나의 프로세서가 두가지 이상의 작업을 동시에 수행할 수는 없기 때문에, 시간을 세분하여 각각의 phase에 한가지씩의 작업을 수행하는 것이다. 근래에 이용되는 마이크로 프로세서를 이용한 시스템들은 거의가 이러한 단계화 임무 시스템에 해당한다. 즉, 하나의 PLC가 한가지의 작업만을 수행하는 것이 아니라, 여러 가지의 입력을 받아 그에 해당하는 결과를 순차적으로 계산해 내는 것이다. 이러한 일련의 작업이 단일의 기능의 구현을 위한 것이 아니고, 전혀 독립된 기능을 단순히 하나의 시스템을 통해 구현하는 것일 경우, 기존의 고장 수목 등의 분석방법으로

신뢰도를 분석하는데는 어려움이 많다.

그래서 이러한 시스템의 분석을 위해서는 주로 마코프 모델을 이용하는데, 일반적으로면서도 다양한 상태를 고려할 수 있다는 장점 때문이다. 그러나 마코프 모델을 이용하기 위해서는 그 적용 대상이 independent process이면서, 직전의 state에만 영향을 받는 process라는 가정이 필요하다. 때로는 이러한 시스템의 확장을 위해서 Semi-Markov process (어떤 state에 머무는 것을 허용하는 마코프 모델)를 이용하기도 하지만, 대상 시스템이 복잡해지면 그 state의 수가 너무 많아지므로 현실적으로 계산이 불가능하며, 각 state간의 transition probability가 특정분포(exponential, normal 등)를 벗어난 분포를 가질 경우에는 처리가 어렵다는 단점이 있다.

이러한 단점을 보완하고자 하는 개량된 방법론들이 연구되고 있는데, Zang 등이 제안한 Binary decision diagram (BDD)을 이용하는 방법과 Gokhale 등이 제안한 이산 사건 시뮬레이션 방법 등이 있다 [36], [40]. BDD를 단계화 임무 시스템에 적용하기 위해서는 phase별로 이용되는 component(또는 모듈)들을 마치 각각의 다른 component가 있는 것처럼 가상하여 BDD를 작성하고, 같은 component끼리의 boolean 연산 부분에 대해서는 별도로 처리하여야 한다. 이 방법을 사용할 경우, 마코프 모델을 이용하는 경우에 비해서 state의 수를 크게 줄일 수 있으므로 연산시간이 빨라져서 실용성이 높아진다. 한편, Gokhale 등이 고장내구성 시스템에의 적용을 목표로 제안한 이산 사건 시뮬레이션 기법을 이용하면 단계화 임무 시스템을 효과적으로 모델링할 수 있을 것으로 판단된다. 기본적으로 이산 사건 시뮬레이션은 대상 시스템에 관계없이 적용할 수 있으므로, 하드웨어로 적용을 확장하여 단계화 임무 시스템을 모델링에 응용할 경우 장점이 많을 것으로 판단된다. 그러나 시스템 내부의 각 모듈간의 관계를 정확히 파악할 수 있어야 한다는 것이 단점이다.

## 제 2 절 BDD를 이용한 단계화 임무 시스템 분석

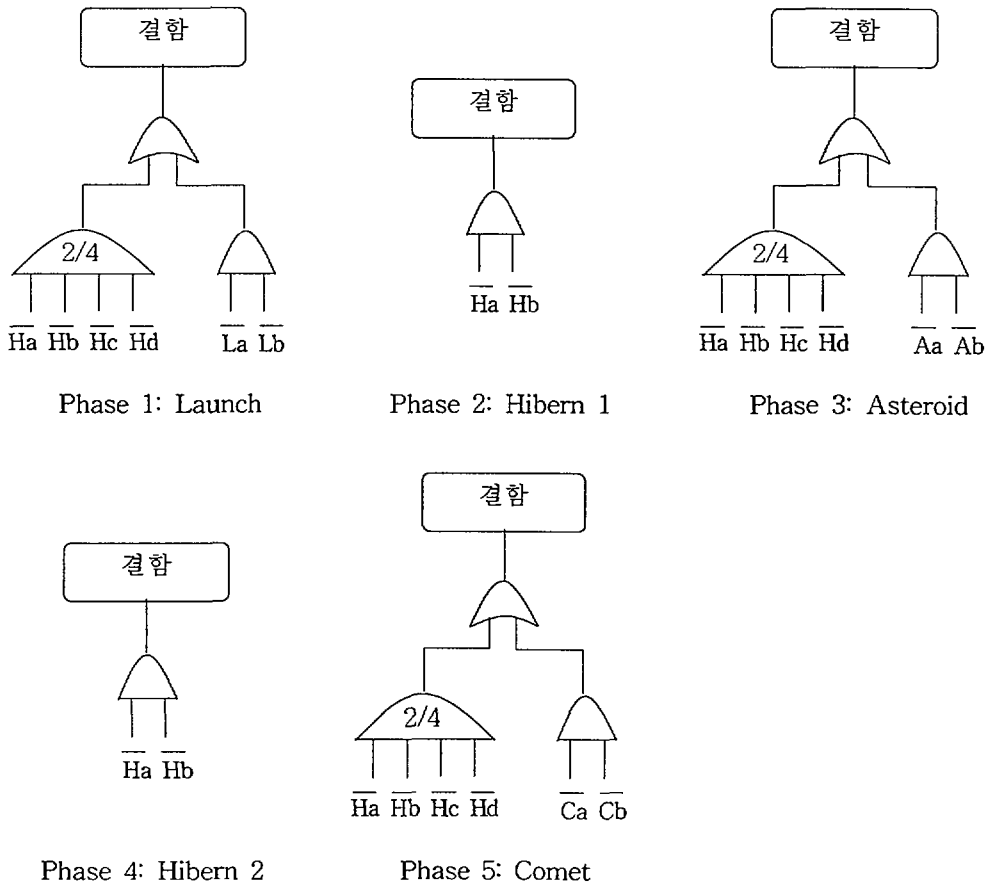
Zang 등은 마코프 모델을 포함한 기존의 모델링 기법들을 이용하여 단계화 임무 시스템을 분석하는 과정에서 비현실적으로 state의 수가 늘어나고 각 state간의 천이 확률을 구하기 어려워지는 등의 비실용성이 나타난다는 점을 개선하기 위해 Binary decision diagram (BDD)를 이용하는 방법론을 제안하였다. [40].

BDD는 Shannon의 decomposition을 이용한 directed acyclic graph (DAG)의 일종이다. Shannon decomposition이란, boolean expression으로 표현된 f를 변수 x의 진위에 관해 분해하는 방법이다. 수식으로 표현하면 다음과 같다.

$$f = x \cdot f_{x=1} + \bar{x} \cdot f_{x=0}$$

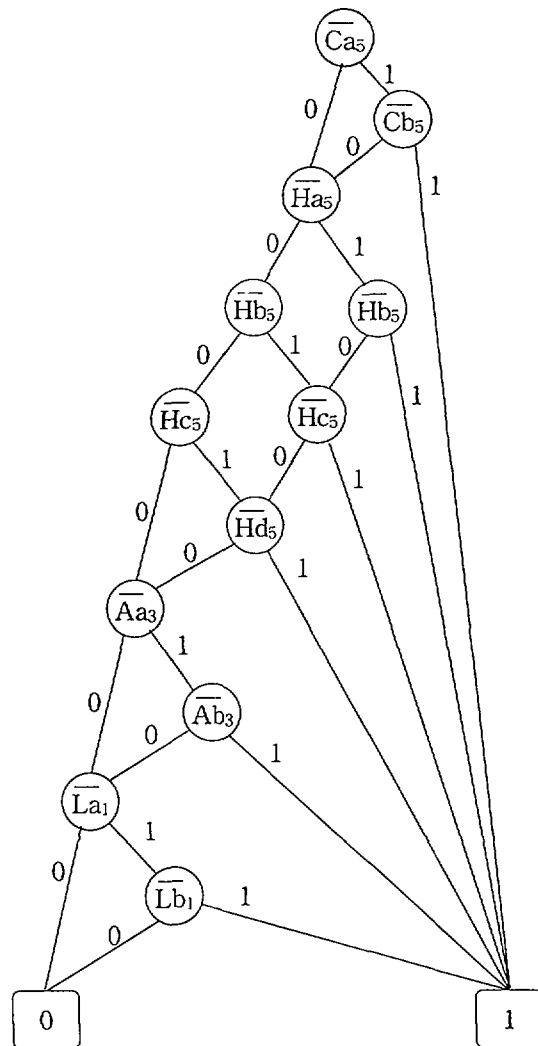
이 방법론을 간단하게 정리하면 다음과 같다.

- 단계화 임무 시스템에서 각 phase에 대해 각각의 다른 component가 있는 것처럼 가상하여 BDD를 작성한다.
- 기존의 BDD에서와는 달리, 단계화 임무 시스템의 경우에는 동일한 component끼리의 연산이 발생하므로, 이를 처리하기 위해 고안된 Boolean algebra를 적용한다.



<그림 4-1 (a)> 5단계로 간략화된 우주선의 시스템 configuration [40]

<그림 4-1> (a)와 (b)는 참고 문헌 [40]의 우주선(space application)에 대한 예제를 나타낸 것이다. 우주선은 Launch, Asteroid, Comet, Hibernation1, Hibernation2의 5가지 operational phase를 가진다. 또한 La, Lb, Aa, Ab, Ca, Cb, Ha, Hb, Hc, Hd의 10개 equipment를 가진다. <그림 4-1 (a)>에 나타나듯이, 모든 operational phase에서 Ha와 Hb가 사용된다. 이러한 단계화 임무 시스템을 BDD를 이용하여 모델링하면 <그림 4-1 (b)>가 된다.



<그림 4-1 (b)> BDD를 이용한 우주선의 신뢰도 모델링 [40]

Zang 등은 BDD를 이용한 분석이 마코프 모델을 이용한 분석과 비교할 때 계산시간이 월등히 향상되는 것을 보였다.

### 제 3 절 이산 사건 시뮬레이션을 이용한 단계화 임무 시스템 분석

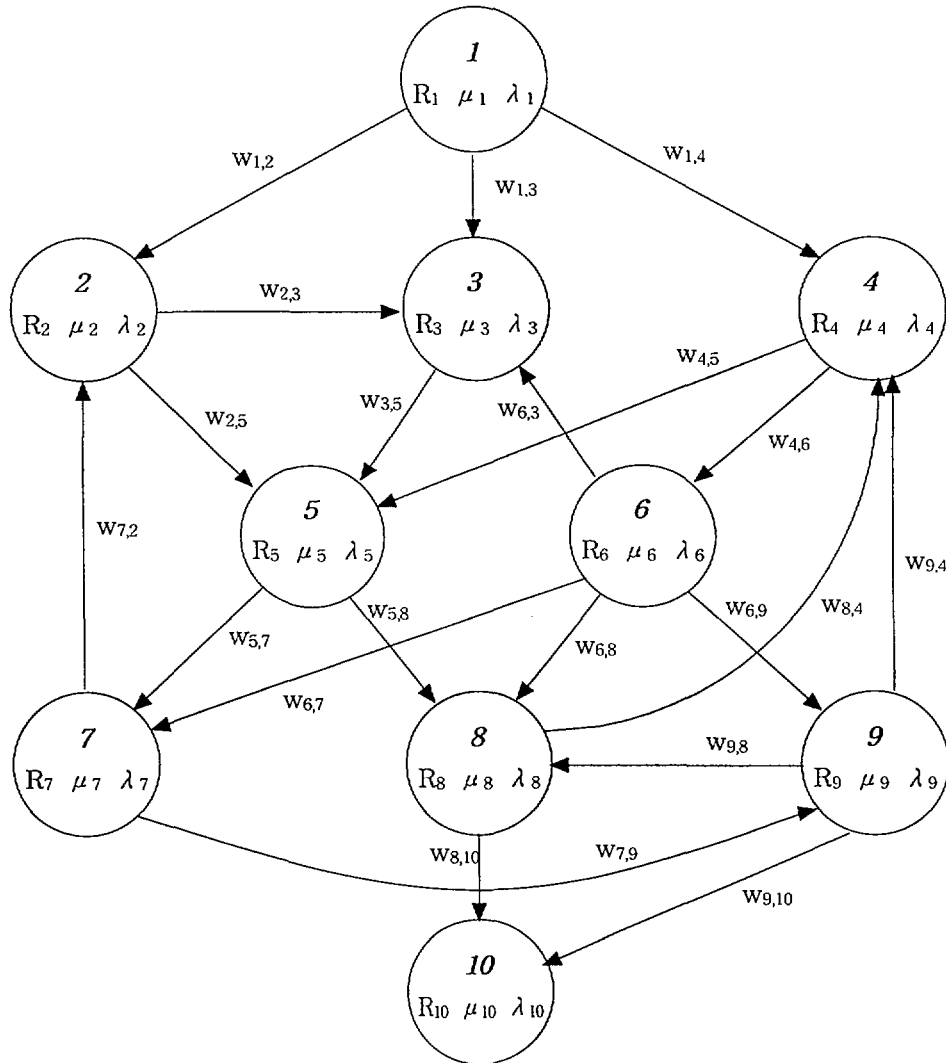
제 3장에서 전술한 바와 같이, 마코프 모델은 복잡한 상황에서는 첫째, 그 크기(state-space)가 지나치게 커져서 현실적으로 다룰 수 없게 될 수 있으며, 둘째, 시스템 내의 각 모듈의 신뢰도 변화나 서로간의 상관관계 등을 고려할 수 없다는 단점이 있다. 단계화 임무 시스템은 이 두가지 문제를 모두 내포하고 있다. Gokhale 등이 제안한 이산 사건 시뮬레이션 기법을 이용하면 이러한 복잡한 상황에서의 시스템을 성공적으로 모델링할 수 있다 [36], [37]. 원저자들은 고장내구성 시스템에의 적용을 목표로 제안한 것이지만, 단계화 임무 시스템에도 응용할 수 있을 것으로 판단된다.

기본적으로 이산 사건 시뮬레이션은 대상 시스템에 관계없이 적용할 수 있다. 그러나 시스템 내부의 각 모듈간의 관계를 정확히 파악하고 있어야 한다는 문제점이 있다. Gokhale 등의 연구에서는 이런 관계들을 모두 정량적으로 파악하고 있다는 가정하에서 진행하였으나, 실제로 이런 자료를 정확히 얻기는 어렵다는 점에 문제가 있는 것으로 파악된다.

Gokhale 등의 연구는 소프트웨어에 국한되어 있으나, 하드웨어로 확장하는 것은 어려운 일이 아닐 것으로 생각되므로, 일반적인 적용성이 높은 이산 사건 시뮬레이션을 적용하여 단계화 임무 시스템을 모델링하는 것의 장점이 많다고 판단된다. 소프트웨어에 적용할 경우, 각 모듈의 실행에 걸리는 시간, 각 모듈이 실행될 확률, 각 모듈의 failure profile 등을 자유롭게 결정할 수 있다. 하드웨어에 대해서도 이러한 값들을 추정할 수 있을 경우 충분히 적용이 가능할 것으로 생각된다. 참고를 위하여 state 1에서 시작하여 state 10에서 끝나는 terminating application의 예제를 소개한다 [36].

<그림 4-2>는 control-flow graph이며, 이것을 토대로 이산 사건 시뮬레이션을 수행하게 된다.  $w_{ij}$ 는 i 모듈의 수행이 끝난 후, j 모듈이 수행될 확률을 나타낸다. 이  $w_{ij}$  값과 각 모듈의 고장률( $\lambda_i$ )과 수리율( $\mu_i$ )이 알려져 있다고 가정하

여  $\Delta t$ 의 시간동안 고장이 일어날 확률을 난수발생을 통해 simulation하는 것이다.



<그림 4-2> 이산 사건 시뮬레이션을 이용한 terminating application의 모델링을 위한 control-flow graph [36].

## 제 5 장 결론 및 활용 방안

디지털 시스템의 정량적 안전성 평가에는 기존의 아날로그 시스템과 크게 다른 방법론을 적용하여야 할 것으로 판단된다. 그 이유는 다음과 같다. 첫째, 디지털 기기에서는 하드웨어의 무작위성 결합 이외에도 소프트웨어 설계 결합으로 인한 결정론적 결합도 고려해야 한다. 둘째, 범용 하드웨어 시스템에 소프트웨어를 통해 기능을 부여하는 방식이므로 특정 기기가 주어진 단일 기능을 수행하는 아날로그와 달라졌으므로 다기능 시스템(phased-mission system)과 공통의 범용 하드웨어나 소프트웨어 코드 사용에 따른 공통원인 고장을 고려해야 한다. 셋째, 디지털 기기의 특성상 병렬 배치나 voting 이외에도 다양한 고장내구성 기능의 구현이 가능해 졌으므로, 이를 안전성 평가에 고려해야 한다.

본 기술 현황보고서에서는 디지털 계통의 특성을 ‘정량적 평가를 수행하기 위한 관점’에서 국내외의 다양한 분야에 적용되고 있는 디지털 기기에 대한 평가 방법론의 현황을 분석하였다. 이와 관련한 부분에 대해서는 국내외를 막론하고 현재 연구가 진행중인 경우가 많고, 일부 사항에 대해서는 논란이 거듭되고 있는 실정이다. 본 보고서는 국내외의 문헌을 통해 디지털 계측제어 계통의 정량적 안전성 분석과 관련된 주요 쟁점들을 3가지로 정리하여 분석하였는데, 첫번째는 소프트웨어와 하드웨어를 분리하여 신뢰도를 추정할 수 있는가 또는 통합하여 추정하여야 하는가에 관한 문제이다. 두번째는 디지털 계통내의 고장내구성(fault tolerance)을 확보하기 위한 메카니즘을 어떻게 신뢰도 분석에 반영할 것인가에 관한 문제이다. 세번째는 단일 계통이 다중의 기능을 수행하는 경우의 신뢰도 분석에 관한 문제이다.

첫번째 문제인 소프트웨어와 하드웨어를 포함하는 PSA 방법론에 대한 국내외의 연구 현황을 분석한 결과, 하드웨어와 소프트웨어가 동시에 포함된 ‘시스템’을 평가하기 위한 방법론들이 활발히 제안되고 있었다. 실제 실험 결과에 의존하여 접근하는 방법과 이론적 모델링을 통해 접근하는 방법으로 대별될 수 있는데, 두가지 경우 모두 고신뢰도 시스템에 적용하기에는 적용성이 낮은 것으로 판단된다. 고신뢰도 시스템의 경우 데이터 수가 부족하여 통계적 처리에 많은 문제가 있을 것으로 판단된다. 그러므로 근래의 연구동향은 대체로 하드웨어와 소프트웨어를 모두 포함한 전체 시스템의 구조와 기능을 정리하고 그 고장 유형(failure mode)에 대한 분석을 수행한 후, 신뢰도 모델을 구성하는 방향으로 정리되고 있다. 이때 하드웨어에 대해서는 무작위성 고장에 중점을 두어 평가하며, 소프트웨



어에 대해서는 설계 및 코딩 결함(design failure & coding error)을 주로 다루게 된다.

두번째 문제인 디지털 계통내의 결함에 대한 내구성을 확보하기 위한 메카니즘에 관한 연구와 이를 어떻게 신뢰도 분석에 반영할 것인가에 관한 연구의 현황을 분석한 결과, 다양한 계층에 다양한 방법으로 적용되고 있는 고장내구성 기법들을 일반적으로 반영할 수 있는 방법론은 아직 개발되지 않은 것으로 파악되었다. 특정한 고장내구성 방법론의 적용에 관한 연구결과들이 발표되고 있으며, 본 보고서에는 이를 정리하여 기술하였다. 가장 확실한 신뢰도 향상 방법은 다중 시스템 여러 종류를 확보하는 것이지만, 보다 적은 비용으로 실현할 수 있는 국지적 기법들도 신뢰도 향상에 효과가 있는 것으로 보고되었다. 하드웨어 차원에서는 적절한 개수의 감시 프로세서를 설치하고, 소프트웨어 차원에서는 N-version 소프트웨어 기법을 활용하는 것이 효과적인 것으로 판단된다. 그러나 이들 방법론을 정량적으로 평가하려는 연구들에서는 유효범위치를 두어 일정 비율의 결함을 감지해 낼 수 있는 것으로 가정하고 모델을 제안하였으므로, 실제 분석 단계에서 이러한 유효범위를 추정해 내는 과정을 정형화하는 문제가 남아 있는 것으로 판단된다.

세번째 문제인 단일 계통이 다중의 기능을 수행하는 단계화 임무 시스템의 경우에 관한 연구들을 분석한 결과, 마코프 모델이 이러한 시스템을 표현하는데 가장 적합한 것으로 파악되었으나, 마코프 모델의 높은 복잡도와 낮은 적용성이 실제 활용에 장애요인이 되고 있었다. 이를 개선하기 위한 연구들이 진행되고 있었으며, 본 보고서에서 BDD 방법론과 이산 사건 시뮬레이션 방법론을 소개하였다. 위의 첫번째와 두번째 문제의 경우 현재까지 공론화된 방법론이 존재하지 않을 정도로 여러 가지 이견이 많은데 비해서, 이 세번째 문제는 비교적 이론적·체계적으로 접근이 가능한 것으로 판단된다.

위에 기술한 여러 가지 문제들 때문에, 현재까지의 정량평가 기술 수준으로는 디지털 기기에 대한 직접적인 정량분석은 그 분석 자체의 신뢰도가 높지 않다. 그러므로 이러한 정량 분석만으로 안전 분석을 수행하는 것은 규제에 관련해서는 허용되지 않으며, 정성적 분석을 병행하여야만 한다. 그러나 디지털 하드웨어 및 소프트웨어의 등급을 결정하는 분류(categorization)에 정량 분석을 유용하게 활용할 수 있다 [41].

한편, 디지털 시스템의 다중성 확보를 위한 분석을 수행하기 위해서는 구체적

인 정량분석이 필수적이다. 공통원인 고장의 가능성이 존재하는 경우에는 전체적으로 어느 정도의 신뢰도를 확보하는 지에 관해서 추정하는 것은, 각 기기에 관한 구체적이고 정확한 정량분석 없이는 불가능하다. 다중성 확보를 위한 노력의 정량적 평가를 위해서는 두가지 측면에서 추후 연구가 진행되어야 할 것으로 판단되는데, 첫째는 디지털 기기에서의 공통원인 고장 발생 원인 및 그 발생 확률을 추정하기 위한 연구를 수행하는 것이며, 둘째는 디지털 기기의 소프트웨어와 하드웨어를 포괄하면서도, 보다 구체적이고 정확한 정량적 안전 분석을 위한 연구를 수행하는 것이다. 본 보고서에서 다루고 있는 부분은 두번째의 연구에 관한 것이며, 첫번째의 연구 내용은 추후 연구되어야 할 것으로 판단된다.

## 제 6 장 참고 문헌

- [1] S.K. Khobare, S.V. Shrikhande, U. Chandra & G. Govindarajan, "Reliability analysis of microcomputer circuit modules and computer based control systems important to safety of nuclear power plants," RESS, vol 59, p. 253-258, 1998.
- [2] 강인수, 조병수, 최문재, "디지털 기반 계측제어 계통의 신뢰도 분석," 한국원자력학회 추계학술발표회 논문집, 1999.
- [3] 윤원영, 윤문원, "MIL-HDBK-217 방법에 의한 원자로 보호 모듈 신뢰도 평가," 한국원자력학회 추계학술발표회 논문집, 1999.
- [4] MIL-HDBK-217F, Military handbook reliability prediction of electronic equipment, United States Department of Defence, 1987.
- [5] Bellcore Standard TR-332, Issue 5, Reliability prediction procedure for electronic equipment, 1997.
- [6] NEA/CSNI/R(97)23, Operating and maintenance experience with computer-based systems in nuclear power plants, 1998.
- [7] W. Bastl & H.W. Bock, "German qualification and assessment of Digital I&C systems important to safety," RESS, vol 59, p. 163-170, 1998.
- [8] Denson et. al., A new system - reliability assessment methodology
- [9] J.G. Choi & P.H. Seong, "Dependability Estimation of Digital system by Operational Profile Based fault Injection," PSA-99, p. 499-506, Washington DC, August 22-26, 1999
- [10] M. Hecht, D. Tang and H. Hecht, "Quantitative reliability and availability assessment for critical systems including software," Proceedings of the 12th Annual Conference on Computer Assurance, Maryland, USA, June 16-30, 1997.
- [11] D. Tang, M. Hecht, X. An & R. Brill, "MEADEP and its application in dependability analysis for a nuclear power plant safety system," IEEE Tr. on Nuclear Science, Vol., 45, No. 3, p. 1014-1021, June 1998.
- [12] D. Tang, M. Hecht, A. Rosin & J. Handal, "Experience in using MEADEP," Proceedings of Annual Reliability and Maintainability Symposium, IEEE, 1999.

- [13] S.R. Welke, B.W. Johnson & J.H. Aylor, "Reliability modeling of hardware/software systems," IEEE Transactions on Reliability, Vol. 44, No. 3, p. 413-418, 1995.
- [14] K.K. Vemuri & J.B. Dugan, "Reliability analysis of complex hardware-software systems," Proceedings of the Annual of Reliability and Maintainability, p. 178-182, 1999.
- [15] J.M. Kaufmann, J.B. Dugan, R. Manian & K.K. Vemuri, "system Reliability Analysis of an Embedded Hardware/Software system using fault Trees," Proceedings of the Annual Reliability and Maintainability, p. 135-141, 1999.
- [16] M. Bouissou, F. Martin & A. Ourghanlian, "Assessment of a safety-critical system including software: A Bayesian belief network for evidence sources," Proceedings of Annual Reliability and Maintainability Symposium, IEEE, 1999.
- [17] D.M. Karydas & A.C. Brombacher, "Reliability certification of programmable electronic systems," RESS, Vol. 66, p. 103-107, 1999.
- [18] A.C. Brombacher, "Maturity index on reliability: covering non-technical aspects of IEC61508 reliability certification," RESS, Vol. 66, p. 109-120, 1999.
- [19] J.L. Rouvroye & A.C. Brombacher, "New quantitative safety standards: different techniques, different results?," RESS, Vol. 66, p. 121-125, 1999.
- [20] 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.
- [21] KAERI/AR-565/00, 소프트웨어 신뢰도의 정량적 평가 기법에 대한 고유 현안 분석, 2000.
- [22] KAERI-CM-110/96, Development of dynamic alarm processing system algorithm and evaluation of alarm system reliability, 1996.
- [23] COOPRA working document, What PRA needs from a digital I&C systems analysis: An opinion, www.coopra.org, 1999.
- [24] Poore, Mills & Mutchler, "Planning and certifying software system reliability," IEEE Software, p. 88-99, January 1993.

- [25] F. Redmill, "IEC 61508: Principles and use in the management of safety," *Computing & Control Engineering Journal*, p. 205-13, October 1998.
- [26] NUREG-0800:HICB-BTP17, "Guidance on Self-Test and Surveillance Test Provisions."
- [27] H. Choi, W. Wang & K.S. Trivedi, "Analysis of conditional MTTF of fault-tolerant systems," *Microelectronics & Reliability*, Vol. 38, No. 3, 1998.
- [28] M. Imaizumi, K. Yasui & T. Nakagawa, "Reliability evaluations of a fault-tolerant system with N watchdog processors," *Transactions of Information Processing Society of Japan* , V.36 N.12, 1995.
- [29] Z. Hocenski & G. Martinovic, "Influence of Software on fault - Tolerant Microprocessor Control system Dependability," *Proceedings of the IEEE International Symposium on Industrial Electronics*, Vol. 3, p. 1193-1197, 1999.
- [30] D. Lantrip & L. Bruner, "General Purpose Watchdog Timer Component for a Multitasking system," *Embedded systems Journal* , Vol. 10, No.4 , 1997.
- [31] NUREG/CR-6463, Rev. 1, "Review Guidelines on Software Languages for Use in Nuclear Power Plant Safety systems."
- [32] A. Mahmood & E.J. MacCluskey, "Concurrent error detection using watchdog processors -A survey," *IEEE Tr. on Computers*, p. 160-174, Vol. 37, No. 2, February 1988.
- [33] M.R. Lyu & V.B. Mendiratta, "Software fault tolerance in a clustered architecture: Techniques and reliability modeling," *Proceedings of the IEEE Aerospace conference*, p. 141-150, 1999.
- [34] J. Wu, E.B. Fernandez & M. Zhang, "Design and modeling of hybrid fault-tolerant software with cost constraints," *J. of systems Software*, vol. 35, p. 141-149, 1996.
- [35] R.K. Scott & D.F. McAllister, "Cost modeling of N-version fault-tolerant software systems for large N," *IEEE Transactions on Reliability*, Vol. 45, No. 2, p. 297-302, 1996.
- [36] S.S. Gokhale, M.R. Lyu & K.S. Trivedi, "Reliability Simulation of

- Component-Based Software systems," Proceedings of the 9th International Symposium on Software Reliability Engineering, p. 192-201, 1998.
- [37] S.S. Gokhale, M.R. Lyu & K.S. Trivedi, "Reliability Simulation of fault-Tolerant Software and systems," Proceedings of Pacific Rim International Symposium on fault-tolerant systems, p. 167-173, 1997.
- [38] Z. Tan, "Reliability and availability analysis of two-unit warm standby microcomputer systems with self-reset function and repair facility," *Microelectronics & Reliability*, Vol. 37, No. 8, p. 1251-1253, 1997.
- [39] P.M. Anderson, G.M. Chintaluri, S.M. Magbuhat & R.F. Ghajar, "An improved reliability model for redundant protective systems - Markov models," *IEEE Tr. on Power systems*, Vol. 12, No. 2, p. 573-578, 1997.
- [40] X.Y. Zang, H.R. Sun & T.S. Trivedi, "A BDD-based algorithm for reliability analysis of phased-mission systems," *IEEE Transactions on Reliability*, Vol. 48, No. 1, p. 50-60, 1999.
- [41] IEC 61838 TR, Ed. 1: Nuclear power plants - Use of probabilistic safety assessment for the classification of instrumentation and control function

# 부록 A COOPRA Digital I&C Working Document 요약

## What PRA Needs From A Digital I&C Systems Analysis: An Opinion

### 확률론적 안전성 평가를 위한 디지털 계측제어 시스템 분석에 관한 의견

#### 제 1 절 디지털 계측제어 시스템 평가 결과의 확률론적 안전성 평가에의 적용

디지털 계측제어 시스템의 평가를 전체 플랜트의 확률론적 안전성 평가에 적용하려면 다음의 3 가지 조건들을 만족하여야 한다.

- 계측제어 시스템이 사고 시나리오에서 차지하는 비중을 다음의 수준까지 정성적으로 모델링할 수 있어야 한다.
  - 계측제어 시스템을 제외한 다른 시스템이 적절히 모사될 수 있는 수준
  - 계측제어 시스템의 설계, 구현, 운영 등에 관한 결정이 내려지고 분석이 될 수 있는 수준
- 시스템 고장을 신뢰성 있는 방법을 이용하여 정량적으로 추정할 수 있어야 한다.
- 다른 시스템이나 운전원에게 영향을 미치는 주요 고장모드(failure mode)를 모두 평가할 수 있어야 한다.

#### 제 2 절 PRA의 framework과 계측제어 시스템에 대한 접근 방법

기존 원자력 발전소의 PRA 형태인 '사고 경위-사건 수목(event tree)', '기능

및 시스템 고장율-고장 수목(fault tree)'의 형태에서는 보조 시스템인 계측제어 시스템의 고장으로 인한 'failure'는 별도의 모델링을 통해 표현하였다. 지금까지의 접근 방법의 문제점은 계측제어 시스템으로 인한 위험도 중 일부만을 모델에 반영하였다는 점이다. 특히 실제 원전에서 빈발하는 '잘못된 신호의 생성으로 인한 운전원의 오동작 유도 가능성' 등은 전혀 고려되고 있지 않다.

### 제 3 절 정성적 모델의 요건

#### (Qualitative modeling requirements)

디지털 계측제어 시스템의 안전 평가가 실제 원전의 PRA에 도움을 주기 위해서는 다음의 두가지 요건을 만족하여야 한다. 첫째는 현재의 원전 PRA 구조에 부합하여야 하며(compatibility), 둘째는 그 구조가 PRA를 기반으로 수행되는 사고 경위 분석을 적절하게 지원할 수 있어야 한다(internal model structure).

- Compatibility: 계측제어 시스템의 모델이 확률적·논리적 모델 구조에서 사용 가능해야 한다는 뜻이다. 즉, 계측제어 시스템의 모델 자체가 사건 수목이나 고장 수목의 형태를 따라야 한다는 뜻이 아니라, 모델의 입력값과 출력값이 이러한 구조에 부합해야 한다는 뜻이다. 입력값으로 시간 관련(time-dependent)한 값이나 연속적인 플랜트 상태값 등을 요구해서는 안되며, 수행 성능에 직접 관련된 시스템의 이산 상태(discrete system state)를 출력값으로 제공하여야 한다.
- Internal model structure: 모델의 상세 정도(level of detail)가 사고 경위 분석을 지원할 수 있는 수준이어야 한다는 뜻이다. 즉, 계측제어 시스템의 오류가 초기 사건으로 작용하는 경우와 안전 기능(safety function)을 저해하는 경우를 구분할 수 있어야 하며, 단일 기능에만 영향을 미치는 경우와 다수의 기능에 영향을 미치는 경우를 구분할 수 있어야 하고, 마찬가지로 기능 수행 실패(loss of function)의 경우와 불필요한 기능 수행(spurious operation)의 경우를 구분할 수 있어야 한다.

### 제 4 절 정량적 모델의 요건

#### (Quantitative modeling requirements)



계측제어 계통에 특별한 정량화 요건이 적용되는 것은 아니나, 정책 결정권자의 판단을 지원할 수 있는 충분한 정확성, 분석의 신뢰성을 확보하여야 하며, 평가의 결과가 내포한 불확실성 수준(uncertainty level)을 추정할 수 있어야 한다.

- 정확성(accuracy): PRA 모델의 정확성에 가장 큰 영향을 미치는 것은 사건(failure event)들 간의 종속성에 대한 처리 방식과 PRA 모델 자체의 완결성이다.
  - 사건들 간의 종속성이 명시적으로 확실히 모델링된 경우에는 사건이 적절히 정량화 되었는지만 확인하면 되지만, 그렇지 못한 경우에는 공통원인 고장(common cause failure)의 확률을 높여주어야 하며, 이를 모델에 적절히 반영하여야 한다.
  - 모델링의 완결성을 보장하거나 증명하는 것은 어려운 일이지만, 적어도 적절한 절차를 통해서 사고 원인들을 처리하였음을 보여주어야만 한다. 그런데 소프트웨어의 경우에는 이러한 단순한 증명에도 어려움이 많다고 알려져 있다.
  - Decision maker의 판단을 지원해야 한다는 PRA의 목적을 상기할 필요가 있다. 즉, '완벽한 결과'가 필요한 것이 아니라 '판단을 내리기에 충분히 정확한 결과'가 필요하다. 이 점을 고려할 때, 다음과 같은 방법으로 기기나 구조 또는 시스템(SSC)이 전체 위험도에 미치는 영향이 미약하다는 것을 보일 수 있다.
    - a. SSC가 포함된 단절집합의 Fussel-Vesely 중요도가 1%이하라는 것을 보인다.
    - b. SSC가 포함된 단절집합들의 빈도가 절삭치내에 있음을 보인다.
    - c. SSC가 포함된 시나리오의 추정치가 기기가 포함되지 않은 시나리오의 추정치보다 현저히 낮음을 보인다. (이때 기기가 포함되지 않은 시나리오의 전체 플랜트에 대한 영향이 작아야 한다)
  - 대개의 경우 단일 SSC의 고장만으로 노심손상이 발생하지는 않으며 다른 SSC의 고장이나 운전원 행위를 포함하기 때문에, 이런 다른 부분에 대한 적절한 정량화(careful attention to the quantification)가 문제의 SSC에 대한 상세 평가의 필요성을 줄여줄 수도 있다.

- 신뢰성(credibility): 확률적 안전평가의 결과를 이용하여 결정을 내려야 하는 정책 결정권자에게 신뢰를 주기 위해서는 정량화에 사용된 자료(data)들의 신뢰성이 확보되어야 한다. 원자력 발전소의 디지털 계측제어 시스템의 경우에는 이러한 자료들을 확보하기가 어렵다는데 문제가 있다. 그러므로 차선의 방법으로 다른 산업계에서의 경험을 차용하여 사용하거나 또는 전문가의 경험을 바탕으로한 자료를 이용한다.
  - Commercial off-the-shelf 하드웨어와 같은 경우에는 타 산업계의 자료가 유용하게 사용될 수 있다. 그러나 소프트웨어의 경우에는, 사용 환경이 달라진 상태에서 기존의 경험 자료를 직접 적용할 수 있는가 하는 문제가 발생한다. 따라서 다른 플랜트의 소프트웨어 고장 자료를 적용하는 경우에는 적용의 당위성(basis for assuming that the data are applicable)을 확보하는 것이 필요하다.
  - PRA에 필요한 자료를 얻을 수 없는 경우나 시간·비용 등의 제약이 심한 경우에는 전문가의 판단을 이용하기도 한다. 이 경우에는 전문가의 질(교육 및 경험의 정도)과 자료를 얻기 위한 과정(질문과 답변)에 유의할 필요가 있다.
  
- 불확실성(uncertainty): 확률적 안전평가 결과의 불확실성은 입력된 모수 자료의 불확실성과 모델 구성의 불확실성의 두가지로 대별될 수 있다. 즉, 자료의 불충분으로 인해 정확한 모수 값을 구하지 못한 경우와 시스템에 대한 이해의 부족으로 인해 적절한 모델링을 하지 못했을 경우에는 불확실성이 높아지는 것이다.
  - 표준이 확보될 때까지는, 모델링 결과에 영향을 주는 주요 가정들을 및 그 가정들의 근거를 명확하게 하는 것이 필요하다.

## 제 5 절 디지털 계측제어 시스템 성능 분석의 이슈

일반적으로 디지털 시스템의 하드웨어 부분에는 기존의 PRA방법론을 직접 적용할 수 있을 것으로 판단되나, 소프트웨어의 경우는 그렇지 못하다. 소프트웨어의 처리에 관한 상반된 견해가 존재하는 두가지 분야에 대해 기술하면 다음과 같다.

- 소프트웨어의 고장률(Software failure rate): 기본적으로 소프트웨어에는 경년열화(aging)가 없다는 점에서 시작되는 문제인데, 소프트웨어의 고장을 기존의 무작위성 모델(aleatory model)로 처리하느냐 지식 모델(epistemic model)로 처리하여야 하느냐 하는 문제이다.
  - 무작위성 모델의 경우 demand rate  $\lambda$ 를 가지는 시스템은  $\lambda(1-p)$ 의 빈도로 성공하고,  $\lambda p$ 의 빈도로 실패한다고 모델링한다.
  - 지식 모델의 경우 demand rate  $\lambda$ 를 가지는 시스템은  $1-p$ 의 확률로  $\lambda$ 의 성공률을 가지거나  $p$ 의 확률로  $\lambda$ 의 실패율을 가지는 것으로 모델링한다.
  - 하드웨어와 소프트웨어를 동시에 사용하기 때문에 문제는 더 복잡해지며(하드웨어나 운전원이 소프트웨어의 고장을 유발하는 입력을 제공하는 경우, 핵심 하드웨어 부품이 열화하여 소프트웨어가 기능을 수행할 수 없는 경우 등), 여러 가지 경우를 고려하면 개략적인 모델링의 경우에는 무작위성 모델을 소프트웨어에 적용하는 것이 합리적이라고 볼 수 있다.
  - 기존의 운전원 모델이나 하드웨어 모델을 참조하면 소프트웨어 모델이 결정론적이어야 할 필요가 없다는 것을 알 수 있는데, 예를 들어, 특정 오류입력에 대한 운전원의 행동이 결정론적이라 할 지라도 그 입력의 생성이 우발적이기 때문이다. 그러나 소프트웨어가 운전원이나 하드웨어보다는 더 결정론적인 성향을 보인다는 것은 명확하며, 그 정도의 문제에 논란의 여지가 있다.
  
- 소프트웨어 고장 자료의 적용(Applicability of software failure data): 소프트웨어 패키지들의 자료를 유사한 다른 소프트웨어 패키지에 적용하여 고장률을 추정할 수 있는가 하는 문제이다. 이것은 약간의 차이로 인해 결과가 완전히 달라지는 소프트웨어의 비선형성(non-linearity)에 기인하는 문제이다. 수집된 자료들을 어떤 방법으로 처리하여 likelihood function for the data를 구하는 것이 적합할 것인지는 소프트웨어의 고장을 우발적(확률적)으로 취급하느냐 결정론적으로 취급하느냐와 밀접한 관련이 있다.
  - 우발적인 모델을 적용하는 경우에는 기존의 하드웨어 고장률에 적용하는 방법론을 별도의 수정 없이 이용할 수 있다.

- 결정론적인 모델을 적용하는 경우에는 새로운 고려가 포함되어야 한다.
- 공통원인 고장(CCF)의 추정이 이와 유사한 경우인데, 이 경우에는 전문가의 지식에 기반한 영향 벡터를 작성하여 이용한다.

## 부록 B National Research Council의 원전 I&C 보고서 6장 번역 · 요약

### Digital Instrumentation and Control Systems in Nuclear Power Plant: Safety and Reliability Issues Chapter 6. Safety and Reliability Assessment Methods

#### 원전 디지털 계측제어 시스템의 안전성 및 신뢰도 현안 제 6 장 안전성 및 신뢰도 평가 방법론

##### 제 1 절 서론

원자력 발전소에 디지털 계측제어 시스템을 적용하는데 있어서는 적절한 방법으로 안전성 및 신뢰성 평가 수행하는 것이 매우 중요하다. 그러한 방법들은 신뢰성 예측, 여유 안전성평가, 정량적 안전 목표치에 의한 성능비교, 절충(trade-off)에 의한 전체적인 안전성 평가 등을 지원할 수 있어야 한다. 이러한 방법들은 충분히 확고해야 하고, 정당함이 증명되어야 하며, 디지털 계측제어 기술이 실제로 공공 안전성을 향상시킬 수 있음을 확신시킬 수 있어야 한다,

##### 1. 현안 사항

원자력 발전소에서 디지털 계측제어 시스템의 안전성과 신뢰성을 평가하기 위한 효과적인 방법론이 필요하다. 이러한 방법론은 잠재적으로 안전하지 못하고 신뢰할 수 없는 응용을 배제시키고, 향상된 안전성과 신뢰성을 활용할 수 있게끔 한다. 디지털 계측제어 시스템의 안전성과 신뢰성을 평가하기 위해 어떤 방법이 사용되어야 할 것인가?

##### 2. 검토 사항

원자력 발전소의 신뢰성과 안전성은 결정론적 기법과 확률론적 기법을 상호 조합하여 평가된다. 이러한 방법들을 디지털 계측제어 시스템의 평가에 어느 정도까지 적용할 것인지와 적절한 적용 방안을 검토한다.

#### 가. 결정론적 평가 방법

설계 기준 사고 분석(design basis accident analysis)은 규정된 사건 시나리오에 대한 발전소 반응의 결정론적 평가이다. 이 분석법은 미국 핵 규제 위원회에 제출된 원자력 발전소 안전성 분석 보고서의 주요 부분을 구성하고 있다.

#### 나. 확률론적 평가 방법

확률론적 위험도 평가(PRA 또는 확률론적 안전성 평가; PSA) 방법은 시스템 수준의 안전성 혹은 신뢰성에 기여하는 사건들의 상대적인 영향을 평가하기 위해 사용된다. 확률론적 방법은 불확실성을 가지는 사건들을 평가하기 위한 통합된 수단을 제공한다. 이러한 분석은 대개 고장 수목 분석을 통해 수행되지만, 사건 수목 분석, 신뢰성 블록도, 마코프 모델 등에 의한 분석 역시 가능하다.

PRA에서 종결사건(end event)의 확률은 고장 사건(failure event)으로 불리는 기본 사건(basic event)들의 발생확률로부터 계산된다. 예를 들어 USNRC는 노심 손상 사건 확률이 원자로 가동 단위 년(year)당  $10^{-5}$ 을 초과하지 않도록 정량적 안전성 목표치를 설정해 놓고 있다. 물론 PRA가 높은 불확실성을 내포하고 있긴 하지만, 중요한 고장 유형의 검색이 가능하고, 단일 시스템이나 부품에 과도하게 의존하지 않는 적절히 균형 잡힌 시스템의 설계를 가능케 한다.

고장수목 분석은 다음의 몇 가지 중요한 목적을 충족시킨다. 첫째, 시스템 고장 거동(behavior)분석과 사고 시나리오의 정확한 기술(document)을 위한 논리적 뼈대를 제공한다. 둘째, 고장수목 모델은 Boolean 대수와 확률의 기본원리를 이용하여 고장사건의 확률계산을 Boolean 논리 함수로 연계시킨다. 즉, 고장수목 모델은 어떤 사건조합들이 종결사건(혹은 정점사건)을 유발시킬 수 있는지 보여줄 뿐만 아니라, 종결사건의 확률이 어떻게 기본사건의 확률로 계산되는지를 정의할 수 있다. 이렇게 고장수목 분석을 이용하여 설계 취약점의 규명, 상대적 위험 평가 등을 수행할 수 있다.

위에서 언급된 것처럼, 안전성과 신뢰성의 확률론적 분석은 고장수목상의 기

본사건들의 발생 확률에 의존한다. 그러나 확률 해석은 사건 확률뿐만 아니라 변화성과 불확실성 역시 표현할 수 있다. 어떤 특정 부품의 고장 가능성을 확률을 이용하여 추정할 수 있지만, 그 추정치는 주변 환경의 확률 분포에 의해 영향을 받는다. 이러한 변화를 나타내는 이차 확률 개념은 점 추정치(point estimate) 주변의 확률분포로 변화성(variability)을 나타낸다. 삼차 확률 개념은 점 추정치의 불확실한 정도를 나타내기 위해 분산(distribution)을 활용한다. 이렇듯 실제로 위험도 평가는 사건 확률, 변화성 확률, 불확실성 확률을 서로 구분하고 있다.

그러나 확률론적 분석의 기본 개념은 사건의확률(event probability)에 있다. 사건 확률은 기본적으로 표본공간의 크기에 대한 사건을 포함하는 부공간의 크기 비율 나타낸다. 사건 확률의 빈도 분석은 가장 일반적인 것으로서, 시도 횟수를  $T$  경우의 시도(trial;  $T$ ) 횟수에 대한 관측 사건(observed event;  $OE$ )의 비율의 극한치로 사건 확률을 정의한다.

$$\lim_{T \rightarrow \infty} \frac{OE}{T}$$

그러나 원자력 분야의 확률론적 안전성 평가시에 고려되는 대부분의 사건들은 극히 드물게 발생하는 사건들로 분류되고, 이는 기본 사건의 발생확률 예측을 매우 어렵게 만든다. 기본 사건의 고장 확률 예측을 위한 몇 가지 데이터 베이스와 편람들, 그리고 원자력 산업계의 사업자 사건 보고서와 기타 편람들을 이용하여 원자력 분야에 이용되는 시스템과 설비에 대한 고장 데이터를 얻을 수 있다.

그러나, 빈도에 관련된 데이터가 아예 없거나 거의 없는 몇몇 기본사건들에 대해서는 확률의 주관적 분석이 이용될 수 있고, 실제로 확률의 주관적 분석만이 이용 가능한 유일한 것 일 수도 있다. 주관적 분석은 일반적으로 사건이 발생할 확신(belief)의 척도로 일컬어지는데, Apostolakis (1990)는 “확률은 확신(belief)에 관한 척도이다. 그러한 확률의 근본적 개념은 ‘보다 가능성이 있는’이다. 즉, 우리는 직관적으로 사건 A가 사건 B보다 더 가능성이 있다라고 말할 수 있다. 확률은 단순히 이러한 가능성에 대한 수치적 표현이다.”라고 표현했다. 그러나 보다 많은 정보를 취득하게 되면, 이를 주관적 분포에 반영하여 현재의 지식 상태에 맞도록 조정할 수 있다.

원자력 분야뿐만 아니라 다른 분야에서도 전문가 판단을 통해 안전성에 관한 확률을 도출하는 경우가 많다. 발생빈도 등의 과거 관측 자료가 주관적 판단에 어떤 영향을 미치는지 Bayesian 분석을 통해 조사할 수 있는데, 실제 자료에 비해 전문가의 판단이 특정한 방향으로 편향되는 경우가 발생하기도 하는 것으로 알려져 있다. 이러한 한계에도 불구하고, 희귀 사상(rare event)의 분석에는 전문가

의 판단 자료가 필요하며, 많은 위험성 계산의 기초 자료가 되어왔다.

전문가들의 판단을 이용한 위해도 분석은 적어도 50여년 동안 효과적인 것으로 입증되었으며 대체 방안(alternative approach)으로 임의 테스트(random test)가 제시되어왔다. 그러나 임의 테스트를 통해 위해 상태를 찾아내는 것은 실제로 견초더미 위에서 바늘을 찾는 것만큼이나 어렵다. 회귀 사건의 시나리오를 이용하여 무작위로(randomly)로 테스트를 생성할 수 있지만, 그러한 시나리오의 기술을 위해서는 역시 풍부한 전문 지식이 필요하다.

### 3. 디지털 시스템에의 적용

디지털 시스템에 대한 결정론적 분석 방법은 원자력 산업에 사용되는 설계기준사고 분석의 일반화로서, 위해도 분석법이나 다른 정형분석법을 포함한다. 디지털 시스템으로부터 기인되는 고장모드를 결정론적 분석기법을 이용하여 분석하는 것은 문제가 되지 않는다. 오히려 디지털 시스템의 확률론적 분석방법에 많은 논쟁의 여지가 있다.

비록 물리적인 결함에 대한 입증된 분석 기법이 존재한다고 할 지라도, 중요 시스템의 설계 결함에 대한 확률론적 분석에는 문제가 있다. 소프트웨어 결함은 그 정의에 의해 '설계 결함'에 해당하기 때문에, 소프트웨어의 확률론적 평가 기법에 논의가 집중될 것이다.

소프트웨어 고장의 발생 여부, 소프트웨어 고장 발생의 무작위성 여부, 소프트웨어 고장률의 개념 유무는 현재 소프트웨어 공학계에서도 논란이 되고 있다. 일부에서는 틀린 결과가 산출되더라도 물리적 변화가 발생하는 것은 아니므로 소프트웨어 고장은 일어나지 않는다고 주장한다. 다른 한편에서는 소프트웨어는 성공적으로 실행되거나 그렇지 않거나 둘중의 하나이므로, 그 신뢰도는 0이거나 1이 된다고 주장한다.

소프트웨어 고장 개념을 수용하는 사람들일지라도 소프트웨어 고장이 확률적으로 모델링될 수 있다는 사실에는 동의하지 않는다. 어떤 이들은 입력과 내부상태가 주어진 상황에서 소프트웨어의 동작은 고정되어있기 때문에 소프트웨어는 결정론적이라고 논하기도 한다. 소프트웨어 고장의 무작위성에 대한 주장은 입력의 불확실성 혹은 임의성에 근거를 두고 있다. 예를 들어, Finelli(1991)는 프로그램이 오류(error)를 일으키게 하는 입력 공간의 영역을 "error crystal"로 규정하고, 소프트웨어 고장은 입력이 "error crystal"로 유입할 때 발생하게 된다고 주



장하였다. 일부 소프트웨어는 신뢰성이 작업 부하(workload)의 함수로 표현될 수 있어, 확률론적으로 모델링될 수 있다는 최근의 실험연구의 결과도 있다.

비안전 소프트웨어의 신뢰성 분석에서는 확률적 분석 기법이 사용되고 있다. 거대 소프트웨어 시스템의 임의 테스트 결과가 소프트웨어 고장 확률에 관한 데이터가 된다는 기본 전제하에, 테스트 결과에 따라 소프트웨어 출시 시기와 테스트를 완결해야 할 시기 등을 결정한다. 이러한 거대 상업적 시스템의 소프트웨어 신뢰성 분석 방법들이 안전관련 중요 시스템에 직접 응용이 가능한 것은 아니다. 소프트웨어 신뢰성 공학에서 사용하는 접근 방법들의 문제점 중의 하나는 아주 낮은 고장 확률( $10^{-7}$  failure/hour 이하)을 확률적으로 검증하기 위해서는 너무 많은 테스트를 수행해야 한다는 것이다.

신뢰성이 매우 높은 소프트웨어는 확률적 분석을 훨씬 어렵게 만들며, 또한 모든 테스트에서 오류를 발생시키지 않을 것으로 기대된다. 만일, 안전관련 중요 소프트웨어(safety-critical software)가 테스트를 통과하지 못했을 경우, 소프트웨어의 에러를 수정하고 테스트를 다시 시작한다. 이런 방식으로 소프트웨어가 많은 테스트를 통과하여야 신뢰성 목표를 달성할 수 있다. Miller(1992)는 임의 테스트(random test)동안 고장을 일으키지 않는 소프트웨어에 대한 고장 확률 예측법을 제안했다. 또한 Bertoino와 Strigini(1996)는 failure-free 테스트 수행시 고장 확률과 프로그램 수정확률을 추정하기 위한 방법을 제안하였으며, Parnas(1990)는 고장확률이 특정 상한 경계(upper bound)값보다 작음을 보이기 위해, 수행되어야 할 테스트의 수를 결정하는 방법을 제안하였다. 이는 NUREG/CR-6113과 유사한 분석법으로서, 이 경우에 안전 시스템의 동작범위는 안전과 불안전 동작 사이의 천이(transition)영역으로 간주된다. 따라서 고장 확률이 주어진 값 보다 작음을 보이기 위해 수행되어야 할 테스트 횟수를 결정하는 공식을 제시하고, 임의 테스트(random test)를 천이영역에서 수행하는 것을 권고한다.

이런 방법의 적용이 타당한지는 수행된 테스트의 속성에 달려있다. 테스트는 실제로 직면하는 입력들을 대표해야 하고, 모든 경계 조건들과 잠재된 모든 위험들을 확실히 내포하고 있어야 한다. 그러나 임의 테스트는 안전성 평가와 품질보증을 위해 적용되는 전체 과정의 한 부분으로 사용되어야 한다. 전체 과정은 정형적 방법론(formal methods) 또는 개발 및 보증 활동 전반에 걸쳐 사용되는 다른 분석 방법들을 포함한다. 테스트와 정형적 분석법들은 서로 상호 보완적인 관계에 있다. 즉, 정형적 분석은 테스트되어야 할 위험한 경우를 결정할 수 있게 해 주으며, 테스트는 분석시 설정된 가정(assumption)들의 입증에 도와줄 수 있

다.

원자력 및 기타 산업에서 활용될 수 있는 고장 데이터가 Paula(1993)에 의해 제공되었다. 비상 정지 계통(Emergency Shutdown System; ESS)의 PLC 고장률은 Mitchell와 William(1993)에 의해 보고되었고, 고장내구성을 가지는 디지털 제어 시스템에서의 고장은 Paula(1993)에 의해 분석되었다. 또한 Paula는 기존의 아날로그 제어 시스템을 고장내구성 디지털 제어시스템으로 교체하는 것이 적절한지를 정량적으로 평가할 수 있는 고장수목 분석 방법을 제안하였다. 운영이력이 90년이 된 시스템의 경우, 279회의 단일채널 고장과 55회의 다중채널 고장이 보고되었다. 55회의 다중채널 고장 가운데 9회는 소프트웨어 결함에 원인으로 인한 것이다. 고장수목 분석은 운전원의 부적절한 행위, 소프트웨어 고장, 외부사건으로 인한 물리적 손상, 유효범위의 부족, 하드웨어의 고장, 통신 고장 등의 고장 유형을 포함한다.

## 제 2 절 현재의 US NRC 규제입장과 계획

USNRC의 사전 승인을 필요로 하지 않는 설비 변경의 요건은 10 CFR 50.59에 기술되어 있다. 변경 승인의 필요 여부는 안전 설비의 오동작 확률이나 사건 발생확률의 증가 여부에 달려있다.

USNRC는 '위험도 정보 또는 성능 기반'의 정책이 강화됨에 따라, 모든 규정의 제정 과정에 PRA의 이용을 확대하고 있다. 그러나 디지털 시스템의 확률론적 분석에 관해서는 현재 USNRC의 규제 입장이 명백히 확립되지 않았다. 1995년 10월 USNRC는 그들의 입장에 대해 "디지털 계측제어 시스템의 신뢰성과 안전성을 보증할 책임은 사업자에게 있다"고 밝혔다.

정성적 소프트웨어 보증 기법이 Lawrence Livermore National Laboratory의 몇몇 NUREG 출판물에 소개되어 있지만, 거기엔 확률론적 분석에 관한 어떠한 논의도 포함되어 있지 않다. USNRC의 입장은 "디지털 시스템에 대한 정량적 신뢰성 평가방법은 표준활용(standard practice)으로 인정될 만큼 충분히 개발되었다고 믿어지지 않는다"는 것이다. 1996년 4월에 이에 관한 토의가 다시 있었으나, USNRC는 10 CFR 50.59의 적용에 있어서 현재의 계산 방법들을 이용한 상대적 발생 빈도 계산이 충분히 의미있는 정확한 것으로 판단되지 않는다고 지적했다.

### 제 3 절 미국 원자력 산업 동향

미국 원자력 산업계에서 디지털 시스템(특히 소프트웨어)에 대한 확률론적 분석은 결정론적 분석법과 혼용되는 추세이다. Paula(1993)가 제안한 고장내구성 디지털 제어 시스템의 고장수목 분석은 소프트웨어 고장을 포함하고 있다. 그렇지만 이러한 분석법은 일반적인 것이 아니며, EPRI에서 제정한 PRA관련 가정과 지침에는 디지털 시스템 고유의 고장 유형이나 소프트웨어에 관한 언급은 없다. 확률론적 분석의 적용에 관해서 산업계는 혼용된 형태를 따르거나 미확정인 입장을 취하고 있다. GE는 개량 비등형 원자로의 설계시, 소프트웨어 고장을 소프트웨어 품질보증과 V&V 방법론을 이용하여 해결하였다. 그러나 Westinghouse AP600의 보호 및 안전 감시 시스템의 PRA에서, 동일 기본 설계를 가지는 소프트웨어 고장에 대한 공통모드 불가용도를  $1.2 \times 10^{-6}$  failure/demand로, 소프트웨어 모듈에 대한 공통모드 불가용도를  $1.1 \times 10^{-5}$  failure/demand로 사용하였다는 것은 주목할 만한 일이다.

### 제 4 절 외국의 원자력 산업 동향

캐나다 원자력 에너지 관리 위원회(Canadian Atomic Energy Control Board; AECB)는 소프트웨어 평가를 위한 새로운 규제 지침을 마련하였고, 그 평가 절차를 정형화하고 있다. AECB의 소프트웨어에 대한 평가는 '요구 명세에 관한 분석', '소프트웨어 개발 및 적용 과정에 대한 체계적인 점검', '소프트웨어 테스트에 대한 분석', '소프트웨어 개발 과정과 관리에 대한 확인'의 4가지 요소에 초점을 맞추고 있다. AECB의 접근 방법은 발전소 안전에 있어서 소프트웨어의 역할을 평가하기 위해 소프트웨어 중요도(criticality) 분석을 필요로 한다. 확률론적 분석은 보호 시스템이 방호해야 할 사고조건을 통계적으로 명백히 제시하기 어려우므로 여기선 필요로 하지 않는다.

영국의 Nuclear Electric사는 Sizewell-B 원자로의 1차 보호 시스템 소프트웨어에 대한 동적 테스트를 수행하고 있다. 신뢰성에 대한 정량화가 인허가에 필요한 것은 아니지만, Nuclear Electric사는 연구, 개발의 일부로 자사 소프트웨어의 신뢰성을 보다 더 정확하게 추정하기 위한 테스트를 계속 수행하고 있다.

### 제 5 절 다른 안전성 중요 산업에서의 동향

다른 안전성 중요 산업에서 결정론적 안전성 분석법이 널리 이용되고 있으며 확률론적 분석법은 혼용되고 있는 추세다. 연방 항공국에서는 소프트웨어 인증을 위해 DO-178B 표준을 사용하며 소프트웨어 고장에 대한 확률론적 평가는 고려하지 않고 있다. 철도 제어 시스템은 안전성평가(요구 분석, 위해도 분석, 고장 유형 및 영향 분석), 추상적 모델링(Petri net, VHDL 시뮬레이션, 마코프 모델), 상세 오류삽입 실험에 기존의 정형적 방법들을 이용하고 있다. 그러나 원자력 산업이 직면하는 문제들이 철도 산업에서도 나타나기 시작하면서 PRA 기반의 분석을 이용하려는 경향을 보이고 있다. 심장박동 관리 시스템의 개발시에도 안전성과 신뢰성 평가, V&V 과정을 각 단계별로 포함시키고 있다.

## 제 6 절 분석

안전성과 신뢰성의 결정론적 분석은 이미 인증되어 있고 디지털 시스템에 적용 가능하다. 정형적 방법은 아직은 광범위하게 사용되고 있지는 않지만, 디지털 시스템의 안전성 분석에 훌륭한 근간을 제공하고 있다 .

디지털 시스템의 확률론적 분석시 세 가지 선택이 가능하다. 첫째, 이용 가능한 데이터와 통계적인 테스트 결과를 이용하여 소프트웨어 고장을 포함한 디지털 시스템의 고장 확률을 예측하는 것이다. 또한 불확실성 분석을 통하여 불확실한 입력에 대한 의존도를 최소화 할 수 있다.

둘째, 소프트웨어의 고장에 관한 가정이다. 즉, 소프트웨어가 고장을 일으키지 않거나 반대로 항상 고장을 일으킨다고 가정하는 것이다. 전자의 가정은 고장 수목 분석에 소프트웨어 고장이 전혀 포함되지 않는 경우와 동일하다. 후자의 경우엔 소프트웨어 고장 확률 1을 할당하고 시스템을 설계한다. 확률론적 소프트웨어 모델링을 주저하는 많은 분석자들은 고장수목 분석시에 소프트웨어 고장을 배제한다. 이것은 소프트웨어가 고장을 일으키지 않는다는 가정과 동일하므로 결과는 지나치게 낙관적이다. 품질보증 기법과 통계적인 테스트를 통해 고장 확률의 적절한 경계값을 결정할 수 있다면, 분석 결과는 보다 더 의미있게 될 것이다.

셋째, 원자력 발전소의 안전성과 신뢰성에 대한 확률론적 분석을 포기하는 것이다. 그러나 이것은 매우 비현실적이다. PRA는 원자력 설비 안전성 분석에 매우 효과적으로 사용되어왔기 때문이다. 만일 기존의 고장수목 분석이 PRA에서 사용된다면, 디지털 시스템 관련 고장모드의 모델링에 한계가 있게 마련이다. 마

코프모델 등의 다른 방법들 역시 사용가능한데, 마코프 모델은 고장내구성 디지털 시스템 분석에 적절한 것으로 알려져 있다. 마코프 모델은 고장수목 모델보다 더 융통성이 있으며, 다양한 의존성, 공통 고장원인, 고장사건 관련성 등의 모델링에 훨씬 유용하다. 그러나 모델을 작성하기가 매우 어렵고 대형 모델을 계산하는데 시간이 많이 걸린다는 단점이 있다.

최근에는 복잡한 마코프 모델의 참조없이 고장수목 분석을 고장내구성 디지털 시스템 분석에 적용하려는 연구가 행해지고 있다. 이러한 동적 고장수목 모델은 플랜트의 다른 시스템의 분석에 이용되는 기존의 고장수목 분석들과 잘 통합될 수 있다.

고장내구성 디지털 시스템은 단일 고장으로 전체 시스템을 down시킬 수 있는 공통 원인 고장의 한 타입인 coverage failure에 매우 민감한 것으로 알려져 있다. Coverage failure는 고신뢰도 시스템의 신뢰성 분석에 큰 영향을 미치는 것으로 보인다. 따라서 coverage failure를 모델링에 포함시키는 것은 매우 중요하다. Paula(1993)는 화학 공정과 원자력 산업에서 사용되는 PLC 시스템의 유효범위 고장에 관한 데이터를 제공하고 있다.

## 제 7 절 결론 및 권고

### • 결론 1 •

결정론적 평가방법(설계기준사고 분석, 위해도 분석, 기타 정형적 분석 절차 포함)은 디지털 시스템에 적용 가능하다.

### • 결론 2 •

소프트웨어 고장확률이 정확히 평가 가능한지 여부, 소프트웨어 고장이 임의 발생하는 것인지 여부는 현재 논쟁중에 있다. 그러나 전체 시스템에 대한 디지털 시스템 고장의 상대적인 영향을 평가하기 위한 PRA 수행시, 소프트웨어 고장 확률을 이용할 수 있다. 원자력 발전소에 대한 PRA 수행시 소프트웨어 고장을 명시적으로 고려하는 것이, 이를 간과하는 것 보다 더 바람직한 결과를 유도해 낼 수 있다.

### • 결론 3 •

소프트웨어 혹은 디지털 시스템에 대해 고장 확률을 할당하는 것은 대다수 회

귀 사건에 대한 확률을 다루는 것과 실질적인 차이가 없다. 소프트웨어 품질 보증 방법은 소프트웨어 고장확률의 경계의 추정치에 대한 근거를 제시한다. PRA의 불확실성 분석과 민감도 분석은 분석결과가 불확실한 모수에 과도하게 의존하지 않는다는 것을 분석자가 확신하도록 해 준다. 다른 PRA에서와 마찬가지로, 소프트웨어 고장확률의 경계에 대한 추정치는 확실한 임의 테스트와 전문가 판단에 의해 얻어진다.

• 결론 4 •

확률론적 분석은 이론적으로는 상용 기기(commercial off-the-shelf; COTS)에 모두 동일한 방식으로 적용될 수 있지만 실제 적용은 어려워질 수 있다. 고장 확률 평가를 위해 현장경험 데이터를 이용할 경우, 기기의 사용 조건이 동일할 수도 있지만 다를 수도 있기 때문이다. 프로그램이 가능한 소자의 소프트웨어 고장 확률은 각각의 응용 사례가 모두 고유한 값을 갖는다. 그렇지만 이런 경우에도 엄격하고 정밀한 테스트를 통해 그 고장 확률의 경계(bound)를 추정하는 것은 가능하다. 오랜 현장경험이 축적된 경우에는 전문가의 판단을 유도할 수 있다.

• 권고 1 •

디지털 시스템에 대한 PRA에 시스템 신뢰성에 관한 소프트웨어 고장의 상대적 영향이 포함되어야 한다.

• 권고 2 •

COTS를 포함한 디지털 시스템의 고장 확률을 예측하기 위한 방법들이 PRA에 사용될 목적으로 개발되어야 한다. 이러한 방법들은 인증 사항, 편람, 사용상의 한계 등을 포함해야 하고, 합리성과 정당성이 입증되어야 한다.

• 권고 3 •

USNRC와 산업계는 자신들의 능력을 평가하여야 하며, 정량적 평가의 한계와 시스템에 요구되는 기능을 디지털로 구현할 수 있다는 확신을 얻기 위한 요건들에 대한 이해를 높이기 위하여 충분한 전문성을 개발하여야 한다.

• 권고 4 •

USNRC는 정량 평가에서의 확신을 증가시키고 불확실성을 감소시킬 수 있는, 디지털 시스템 분석에 대한 진보된 기술을 개발하기 위한 프로그램을 지원하는 것을 고려해야 한다.

서 지 정 보 양 식							
수행기관보고서번호		위탁기관보고서번호		표준보고서번호		INIS 주제코드	
KAERI/ - /							
제목 / 부제		확률론적 안전성 평가에서의 디지털 계측제어 계통 고유 현안 분석					
연구책임자 및 부서명 (주저자)		강현국 (종합안전평가팀)					
연구자 및 부서명		성태용 (종합안전평가팀), 임홍섭 (종합안전평가팀), 정환성 (하나로운영팀), 박진희 (종합안전평가팀), 박진균 (종합안전평가팀), 이기영 (동력로기술개발팀), 박종균 (동력로기술개발팀)					
출판지	대전	발행기관	KAERI		발행년	2000.2.	
페이지	78 p.	도표	있음(○), 없음( )		크기	21×29.7cm	
참고사항							
비밀여부	공개(○), 대외비( ), — 급비밀		보고서종류		기술현황분석보고서		
연구위탁기관				계약 번호			
초록		<p>디지털 시스템의 정량적 안전성 평가는 하드웨어의 우발성 결함 이외에도 소프트웨어 설계 결함으로 인한 결정론적 결함도 고려해야 하므로 기존의 아날로그 시스템의 경우와 크게 달라져야 할 것으로 판단된다. 본 보고서는 국내외의 문헌 조사를 통해 디지털 계측제어 계통의 정량적 안전성 분석과 관련된 주요 쟁점들을 다음과 같이 3가지로 정리하여 분석하였다.</p> <ol style="list-style-type: none"> <li>1. 소프트웨어와 하드웨어를 통합하여 안전성 및 신뢰성을 추정하는 방법론 하드웨어와 소프트웨어가 동시에 포함된 '시스템'을 평가하기 위한 방법론들이 활발히 제안되고 있었으며, 대체로 하드웨어에 대해서는 무작위성 고장에 중점을 두어 평가하고, 소프트웨어에 대해서는 설계 및 코딩 결함을 주로 다루게 된다.</li> <li>2. 디지털 계통내의 고장내구성을 확보하기 위한 메카니즘을 신뢰도 분석에 반영하기 위한 방법론</li> <li>3. 단일 계통이 다중의 기능을 수행하는 경우의 신뢰도 분석을 위한 방법론</li> </ol>					
주제명키워드 (10단어내외)		디지털 시스템 신뢰도, 확률론적 안전성 평가, 소프트웨어, 고장내구성					



## BIBLIOGRAPHIC INFORMATION SHEET

Performing Org. Report No.	Sponsoring Org. Report No.	Standard Report No.	INIS Subject Code
KAERI/			
Title / Subtitle	A Technical Survey on Issues of the PSA of Digital I&C Systems		
Project Manager and Department	H.G. Kang (Integrated Safety Assessment team)		
Researcher and Department	T.Y. Sung (ISA team), H.S. Eom (ISA team), H.S. Jeong (Hanaro) J.H. Park (ISA team), J.K. Park (ISA team), K.Y. Lee (ARTD team), and J.K. Park (ARTD team)		
Publication Place	Taejon	Publisher	KAERI
			Publication Date
			2000.2.
Page	78 p.	Ill. & Tab.	Yes( <input type="radio"/> ), No ( <input type="checkbox"/> )
			Size
			21× 29.7cm
Note			
Classified	Open( <input type="radio"/> ), Restricted( <input type="checkbox"/> ), ___ Class Document	Report Type	Analysis Report
Sponsoring Org.		Contract No.	
Abstract (15-20 Lines)	<p>This report describes the review results of the safety assessment and reliability analysis techniques of digital instrumentation and control (I&amp;C) systems. The techniques are far from that of analog I&amp;C systems because of the characteristics of digital systems. This report categorizes the current issues related to the safety assessment of digital I&amp;C systems into three groups as follows:</p> <ol style="list-style-type: none"> <li>1. The methodologies which could integrate the characteristics of hardware and that of software</li> <li>2. The methodologies which effectively represent safety improvement due to the fault-tolerant mechanisms embeded in digital I&amp;C systems</li> <li>3. The methodologies which could effectively represent the phased-mission systems</li> </ol>		
Subject Keywords (About 10 words)	Digital system reliability, Probabilistic safety assessment, Software, Fault-tolerance		