IT 시스템 장애탐지력 향상을 위한 하이브리드 모니터링 프레임워크 개발

박석환*, 손지성**, 백두권**†
*고려대학교 컴퓨터정보통신대학원 소프트웨어공학과,
**고려대학교 정보통신대학 컴퓨터□전파통신공학과
*psakai@korea.ac.kr, **{redfunky07, baikdk}@korea.ac.kr

Development of hybrid monitoring framework for fault detectability enhancement on IT-System

Sok-Hwan Park*, Jiseong Son**, Doo-Kwon Baik**†
*Dept. of Software Engineering, Korea University
**Dept. of Computer and Radio Communications Engineering, Korea University

요 약

본 논문은 다양한 기기로 구성된 분산 IT 환경에서 보다 신속한 장애 탐지 및 원인 파악을 위한 Monitoring framework 을 제안한다. 제안된 Framework 은 전통적인 Event Correlation 기법과 전통적 기법의 문제점 해결을 위한 Probing 기법을 혼용한(Hybrid) 것인데, 이를 통해 각각의 기법이 지닌 한계를 극복하고, 장애탐지 능력을 향상시킨다. 즉, Event Correlation 기법은 임계영역에서 장애 판별이 어려운 문제점이 있는데, 이러한 경우 장애 여부를 확신하기 위해 Probe station 을 이용하여 Probe 를 실행함으로써 장애 판단 능력을 향상시켰다. 또한 Probing 기법은 복잡한 분산 IT 환경에서 Probe Set 를 구성하는 어려움이 있는데, 이를 회피하는데 도움을 준다. 본 논문에서는 Web 기반 애플리케이션에서 User-Session 을 기반으로 추출한 Http Packet 를 Probe set 으로 이용하여 구현하였다.

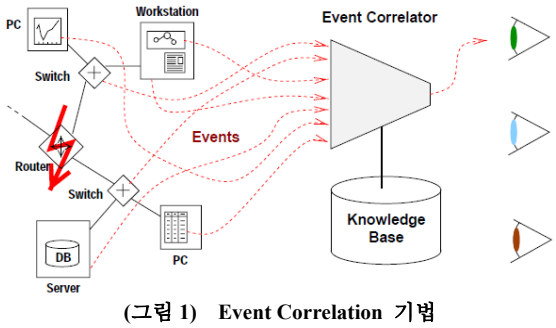
1. 서론

비즈니스의 IT 의존도는 점차 증가되어 금융 거래의 98% 이상이 온라인을 통해 이루어 지고 있다. 이에 따라 IT 시스템에 발생한 장애는 비즈니스 불편을 초래함은 물론 때론 비용 및 인명 손실로 이어지기도 한다. 많은 조직들은 이러한 IT 시스템의 중요성을 인식하여 시스템 장애(Fault)를 조기에 탐지하고, 피해를 최소화하기 위해 IT-장애 통합관제 시스템을 도입 또는 구축하여 운영하고 있다.

하지만, IT-장애 통합관제 시스템을 운영하는 조직들은 여전히 시스템이 감지하지 못한 장애로 인해 비즈니스에 심각한 영향을 받고 있다. 또한 시스템을 구성하는 각종 장치로부터 발생하는 수많은 이벤트를 효과적으로 필터링(Filtering)하고 정확히 해석하여 장애 여부를 판단하는데 한계가 있다. 관제 시스템의 대응 능력을 향상시키기 위해서는 모니터링 대상을 확대하고, 측정된 데이터의 해석능력을 향상시켜야 하는데, 이는 추가적인 비용을 요구한다[1]. 또 다른 접근 방법은 Probe 라 불리는 일종의 테스트 요청을 관제 대상 시스템에 보내고, 그 결과를 장애 판단에 활용하는 방법이다[2]. 이러한 기법은 기존 방법에 비해 획기적인 비용 절감이 가능하나, 환경이 복잡해지면 Probe-set 구성의 어렵고, 둘 이상의 복합 장애에는 사실상 대응이 불가능 하다는 한계를 가지고 있다. 따라서, 본 논문에서는 각종 장치에서 수집한 데이터와 조건만으로 판단이 어려운 경우 Trigger 를 발생하여 추가적으로 Probe 를 실행시켜 장애 여부를 확인할 수 있게 하는 하이브리드 모니터링 프레임워크를 제안한다.

2. 관련연구

IT 시스템은 다양한 기기와 S/W 로 구성되며, 이 중 어느 하나라도 제 기능을 발휘하지 못하면 장애가 발생한다. 이러한 장애를 조기에 탐지하기 위한 대표적인 접근법에는 Event Correlation 과 Probing 이 있다. 먼저 Event Correlation 은 그림 1 과 같이 다양한 관측 대상으로부터 주기적으로 데이터를 수집하고, 해석 엔진(Event Correlator)에서 사전에 등록한 조건 등을 통해 장애 여부를 판단하는 체계로 구성된다.



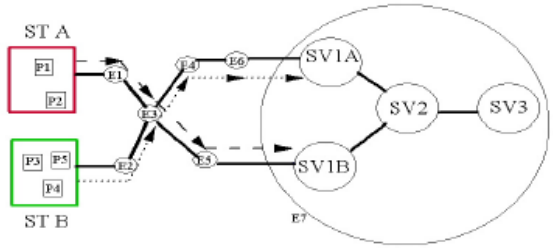
(그림 1) Event Correlation 기법

†: 교신저자 (백두권)
이 연구에 참여한 연구자는 '2 단계 BK21 사업' 의 지원을 받았음

Event Correlation 은 관측 대상 자원이 증가됨에 따라 수집되는 이벤트의 양이 급속히 증가하는 문제가 발생하며 이를 Event Bursts 라 한다. 이러한 수많은 이벤트 중에서 주요한 이벤트를 가려내고 이벤트간 상호 관계를 해석하기 위한 연구가 제안 되었다. 하지만, Event Correlation 기법은 기존의 많은 연구에도 불구하고, 여전히 아래와 같은 문제점을 가진다.

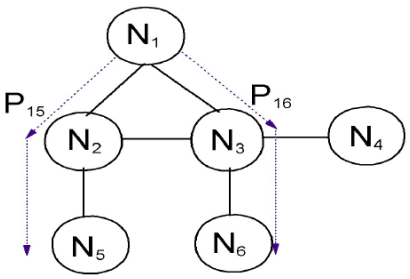
- **Event Correlaton 기법의 문제점**
  - 이벤트 수집을 위한 별도 장치 필요
  - Event Correlator 설계 및 관리 어려움
  - 인프라 변경 시 기존 모델 이용 불가

또 다른 접근방법은 Probe 라는 일종의 테스트용 요청을 이용하여 장애 여부를 탐지하는 것인데, 대표적인 것이 Ping 테스트 이다. 그림 2 와 같이 Probe Station A, B 로부터 Probe 를 발생시켜 각각의 N/W 경로를 통해 SV1B-SV2-SV3, SV1A-SV2-SV3 로 보내고, 응답 결과를 통해 장애탐지 및 발생지점을 파악한다.



(그림 2) Probing 기법

Probing 기법은 적은 비용으로 장애를 쉽게 탐지 할 수 있다는 장점이 있으나, 환경이 복잡해질 경우 장애탐지 및 발생지점 파악을 위한 Probe-set 구성이 매우 어렵다는 문제가 있다. 기존 연구는 Pre-Planned 기법과 Active Proving 기법이 대표적이다[2][3]. Pre-Planned Probing 의 경우 장애탐지 및 발생지점 파악을 위한 Probe-set 를 사전에 구성하는 방법인데, 이 경우 많은 Probe-set 이 필요하다. 그림 3 는 Node 가 여섯 개이고, Probe Station 이 두 개인 N/W 환경을 예시로 하고 있다. N_1 ~ N_6 은 검사 대상 Node 이며, P_{15} 는 Node1 을 시작으로 Node5 로 끝나는 Probe 이다. Probe Station N_1, N_4 로부터 생성되는 Probe 를 구해보면 총 9 개의 Probe 가 생성 가능한데, 모든 Probe 을 운용하는 것은 비효율적이다. 따라서, 장애탐지 및 발생지점 파



(그림 3) Node 6, Probe Station 2 환경

악에 필요한 최소의 Probe 는, 표 1 과 같이 P_{15}, P_{16}, P_{42} 로 Node 와 Probe 간 관계가 직교분할로 구성되어야 한다[2].

<표 1> 근본 원인 파악이 가능한 최소 Probe

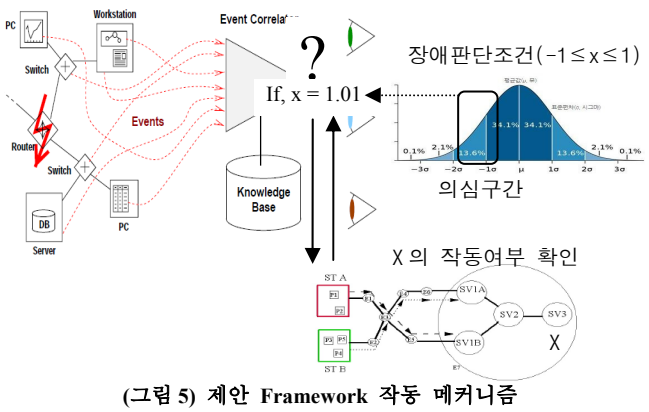
구분	N_1	N_2	N_3	N_4	N_5	N_6
P_{15}	1	1	0	0	1	0
P_{16}	1	0	1	0	0	1
P_{42}	0	1	1	1	0	0

Active Proving 기법은 그림 4 과 같이 장애탐지와 발생지점 파악 단계를 구분하여, 최소의 Probe-set 으로 장애를 탐지한 뒤 각각의 장애 상황에 맞춰 발생지점 파악을 위한 Probe 를 동적으로 구성하는 방법이다. Pre-Planned 와 비교하여 Probe 의 개수를 약 67~72% 감소 시킬 수 있다[3]. 하지만, 동적으로 Probe 를 구성하는데 추가적인 비용이 필요하다는 문제도 있다. 실제 상황에서는 Node 구성이 보다 복잡하여, Probe set 를 산출하는데 NP-hard 문제가 있다. 따라서, 모든 경우의 수를 고려하지 않고 Greedy 나 Quick Search 등 Approximation 알고리즘을 사용하여 단순화 하는 것이 일반적이다[3]. 또한, Probing 기법의 경우 단일 지점 장애(Single Point of Failure)에는 대응이 가능하지만 복합 장애의 경우 사실상 대응이 불가능한 문제도 있다.

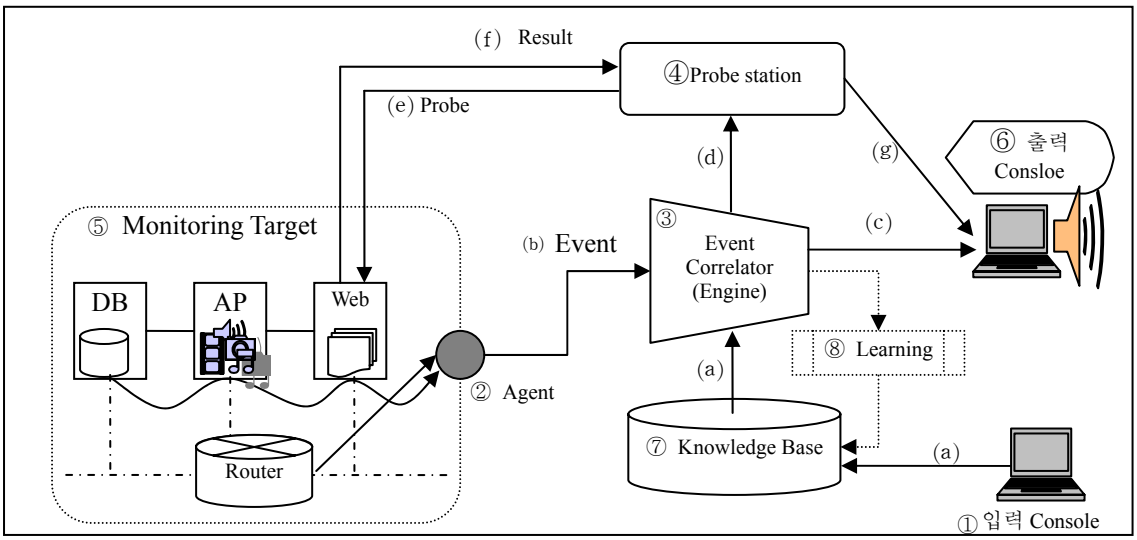
(그림 4) Active Probing

3. 제안 Framework

그림 5 는 본 논문에서 제안하는 Framework 의 작동 메커니즘이다. 모니터링 대상 장치에서 수집한 데이터와 Knowledge Base 에 저장된 판단 조건으로 장애여부 판단이 어려운 경우, 예를 들어 판단 조건이 -1 ≤ X ≤ 1 이고 측정값이 1.01 이라면, Trigger 를 발생시켜 X 를 포함하는 장치 또는 서비스에 Probe 를 실행시켜 그 결과를 통해 장애 여부를 확인하는 방법이다.



(그림 5) 제안 Framework 작동 메커니즘



(그림 6) 제안 Framework

그림 6 은 본 논문에서 제안하는 Framework 이다. Framework 의 주요 구성은 입출력 Console, 정보수집 Agent, Event Correlator, Probe Station, Monitoring Target 으로 되어있다. 각 구성부의 기능 및 역할을 살펴보면, 사용자는 입력 Console①을 통해 Knowledge Base⑦에 Rule(a)을 등록하고 관리한다. Agent②는 주요 자원의 현재 상태를 주기적으로 점검하여 Event(b)를 Event Correlator③로 전송한다. Event Correlator③는 수집된 Event(b)와 추론 또는 경험으로 얻은 Knowledge Base(a) (Threshold 또는 Rule 등)를 비교해서 장애 여부를 판단하고, 결과(c)를 출력 Console⑥로 보낸다. 이때 수집된 Event(b)와 Knowledge Base(a)로 명확하게 판단할 수 없는 경우 Trigger(d)를 발생시키다. Probe Station④은 Trigger(d)에 따라 Probe(e)를 수행하고, Result(f)를 받아 장애 여부를 판단하여 결과(g)를 출력 Console⑥로 보낸다. Probe Station④은 Trigger(d)에 의해서 작동하지만 향후에는 중요 업무에 대해서는 Trigger(d)없이 작동 가능하도록 하기 위해서 판단 기능을 부여하였다. Knowledge Base⑦에는 Learning⑧ 모듈을 두었는데 이는 향후 발전을 위한 것이며, 본 논문에서는 구현하지 않았다.

4. 구현

제안된 Framework 구현 환경은 다음과 같다.
1) 모니터링 서버 : HP Unix, 2) Knowledge Base 저장소 : Oracle 11g, 3) Agent 툴 : Tivoli 및 Maxigent, 4) 개발언어 : Java (JDK 1.5), 5) Probe Station 플랫폼 : Window 기반 PC, 6) Http Packet 캡쳐툴 : Fiddler, 7)모니터링 대상 환경 : Web 기반 N-Tier (A 보험사)

4.1 시스템 구성

제안된 Framework 구현을 위해 기존 Event correlation 방식에서 운영되는 Knowledge Base 에 '장애의심구간정보' 및 측정값이 장애의심구간일 경우에 실행될 'Probe set 정보'를 추가하였다. 또한, Event Correlator 에는 측정값이 장애의심구간일 경우 Probe Station 에 Probe 의 실행을 요청하는 Trigger 기능을 추가하였다. 마지막으로 Probe Station 은 Trigger 에 따라 Probe 를 실행하고, 결과를 받아 장애 여부를 판단할 수 있도록 하였다.

4.2 모니터링 대상 선정

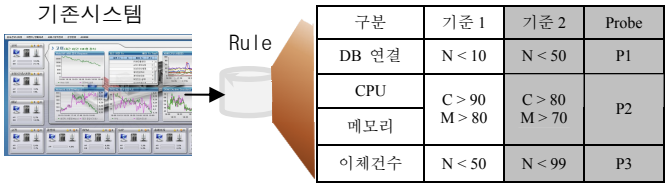
표 2 와 같이 모니터링 대상 환경의 종속성 관계를 도출하여 우선 순위가 높은 자원을 모니터링 대상으로 선정하였다. 예) nRouter 의 경우 nSwitch 와 hWS, hAP 에 1 차 연관이 있고, hDB 및 모든 자원에 2 차 또는 이상의 연관이 있으므로 우선 순위가 높아 선정함

<표 2> 모니터링 대상 환경의 종속성

구분	Dependeny
Network	nRouter → nSwitch
서버(H/W)	hWS hAP hDB
미들웨어	sWS sAP sDB
End User 서비스	pA pB pC* oD* bE bF

4.3. 장애판단기준 및 Probe set 설정

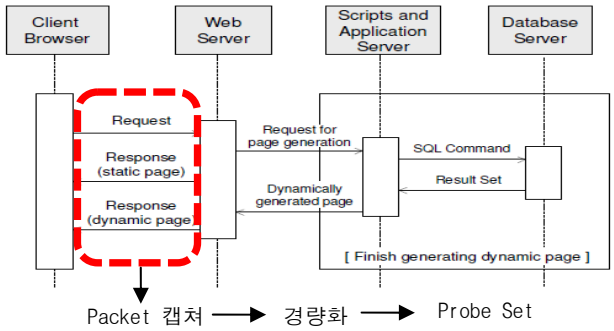
4.2 에서 선정한 모니터링 대상에 그림 7 과 같이 장애판단 기준을 각각 기준 1, 기준 2 로 세분화하여 설정하고, 기준 2 의 경우 추가적으로 Probe 를 실행하여 장애 여부를 확인하도록 하였다. 즉, 모니터링 결과 평일 영업시간에 DB 연결이 50 미만으로 수집된 경우 장애로 의심되며, 이 경우 종속성 관계를 통해 미리 정의된 Probe 를 자동으로 수행하도록 하였다. 해당 Probe 는 장애가 의심되는 DB 와 연결이 필요한 것을 선정하여 Probe 의 실행 결과를 통해 장애 판단이 가능 하도록 하였다.



(그림 7) 장애판단 기준 및 Probe set 설정

4.4. Probe set 생성

4.3 에서 정의한 기준 2 에 따라 모니터링 시스템에 필요한 Probe 가 선정되면, 그림 8 과 같이 사용자 PC 와 웹 서버가 주고받는 Http Packet 를 Fiddler 라는 툴을 사용하여 Capture 한 뒤 불필요한 이미지 정보 등의 제거하여 Probe set 으로 구성하였다 [4-7].



(그림 8) Probe set 제작과정

5. 평가

본 논문에서 제시한 Framework 와 기존 기법을 비교한 자료는 아래 표 5 와 같다. 임계값 설정 비용 및 변경 비용, 필요한 Probe Set 규모, 알고리즘의 복잡도를 비교 항목으로 선정하였다. 제안된 Framework 는 장애 여부를 판단하는데 있어, 장애 판별이 용이한 구간은 수집된 데이터만으로 판단하고, 장애가 의심되어 확인이 필요한 구간은 Probe 를 실행하여 확인한다. 따라서, 기본 방법에 비해 임계기준 설정이 용이하다. 또한 Probe set 으로 사용한 Http Packet 의 경우 인프라 및 User Interface 변경에 대한 독립성이 비교적 높다. Probe 는 특정 조건을 만족하는 경우에만 제한적으로 실행되는 단순한 알고리즘으로 구현되며, 필요한 Probe set 의 규모도 기존 대비 현저히 적다는 장점을 가지고 있다.

<표 5> 제안 Framework vs 기존 기법간 비교

구분	임계값 설정비용	변경 비용	필요 Probe Set	복잡도
Pre-planned probe	N/A	50	100	50
Active Probing	N/A	50	30	100
제안 Framework	**50**	**30**	**10**	**30**
Event correlation	100	100	N/A	100

* 100 점 척도로 정성평가함

6. 결론 및 향후 연구

본 논문에서는 IT 시스템 장애 탐지력 향상을 위해 기존 Event correlation 기법이 가지고 있는 Event Bursts 문제와 장애여부 판단의 어려움을 해결하고자 하였다. 이를 위해 시스템 자원간의 종속성 관계를 기반으로 모니터링 대상을 최소화하였고, 장애 여부 판단에 사용되는 기준인 임계값 설정 방법을 단순화 하였다. 또한, Probing 기법을 Event correlation 으로 판단하기 어려운 상황에 대해서만 보조수단으로 활용되도록 하여, Probe set 구성 및 작동 알고리즘을 단순화 하였다. 즉, 장애 판단을 위한 임계값 경계에서는 장애 여부 판단을 돕기 위해 Probe 를 실행함으로써 장애 발생 여부를 확인하도록 하였다. 이를 통해 기존 대비 임계값 설정이 용이하고, 시스템 구성 변경 시 대응력이 향상 되었다. 또한 Probing 기법 대비 필요한 Probe set 의 규모가 현저히 감소되었다.

본 논문에서 제시한 Probing 은 제한된 조건에서만 수행되며, Probe set 이 사전에 준비되어야 한다. 향후 연구에서는 Trigger 조건에 따르는 Probe set 를 동적으로 구성하여 보다 유연한 Framework 으로 발전시킬 필요가 있다.

참고문헌

[1] B. Gruschke. “Integrated Event Management : Event Correlation Using Dependency Graphs.” Proc. of 9th IFIP/IEEE Int'l.Workshop on Distributed Systems Operation & Management (DSOM98), 1998.

[2] M. Brodie, I. Rish, S. Ma, A. Beygelzimer, and N. Odintsova. “Strategies for Problem Determination Using Probing. Technical report”, IBM T.J. Watson Research Center, 2002.

[3] I. RISH, M. BRODIE, N. ODINTSOVA, S. MA, G. GRABARNIK. “Real-time Problem Determination in Distributed Systems using Active Probing”, IEEE/IFIP (NOMS), (Seoul, Korea, 2004).

[4] S. Sampath, V. Mihaylov, A. Souter, L. Pollock. “Composing a framework to automate testing of operational Web-based software”, Proceedings of the 20th International Conference on Software Maintenance, IEEE Computer Society Press, Los Alamitos, CA, 2004

[5] Sara Sprenkle, Emily Gibson, Sreedevi Sampath, Lori Pollock. "Automated Replay and Failure Detection for Web Applications", International Conference on Automated Software Engineering(ASE), November 2005

[6] Sreedevi Sampath, Renee Bryce. “Improving the effectiveness of Test Suite Reduction for User-Session-Based Testing of Web Applications”, Elsevier Information and Software Technology Journal, 54(7):724-738, July 2012

[7] Sreedevi Sampath, Sara Sprenkle, Emily Gibson, Lori Pollock, and Amie Souter Greenwald. “Applying Concept Analysis to User-session-based Testing of Web Applications”, IEEE Transactions on Software Engineering, Vol. 33, No. 10, pgs 643 - 658, October 2007. Copyright 2007 IEEE.

[8] Chan Uk Jin. “Design the Management of Enterprise IT System using Failure Pre-Detection”, Dong-Eui University, 2006