유비쿼터스 컴퓨팅 환경의 자동적인 결함 허용 기법 설계

최창열 김성수
아주대학교 정보통신전문대학원
{clchoi, sskim}@ajou.ac.kr

A Design of Autonomic Fault Tolerant Mechanism in Ubiquitous Computing Environment

Changyeol Choi Sungsoo Kim
Graduate School of Information and Communication, Ajou University

요 약

유비쿼터스 컴퓨팅은 네트워크로 상호연결된 프로세서로 구성되며, 하나 이상의 컴퓨터로 이뤄진다. 하지만 관리 시스템의 규모 및 자원의 양이 방대하여 이를 모두 사용자 및 관리자가 관리하는 것은 역부족이다. 이에 대한 대안으로 인간의 자율 신경계처럼 시스템 스스로 주위상황 변화에 대응할 수 있는 능력을 부여하자는 오토노믹 컴퓨팅에 대한 연구가 활발히 진행되고 있다. 따라서 본 연구에서는 사람의 간섭을 최소화하면서 예기치 못한 결함이 발생하여도 서비스를 지속적으로 제공할 수 있는 능력을 시스템에 부여하기 위한 자동적인 결함 허용 기법을 제안하고 설계한다.

1. 서 론

유비쿼터스 컴퓨팅은 사람, 컴퓨터와 사물이 통합된 환경에서 기능의 최적화를 위해 이질적인 시스템의 연계성을 제공하고자 하는 것이다[1]. 따라서 5 Any (Anytime, Anywhere, Anything, Anynetwork, Anydevice)화를 달성하여 사용자가 원하는 서비스를 제공받을 수 있어야만 한다. 이를 위해 현 시스템 개발은 이질적 환경을 통합할 수 있는 기술 개발에 초점을 맞추고 있는데, 대표적인 예는 OSGi, J2EE, Eclipse 등과 같은 플랫폼이다. 하지만 대다수 연구가 기능적 통합 및 연결성(functional integra-tion and connectivity)에만 치중하여 기술을 개발하고 있어서 예기치 못한 결함이 발생하여 서비스 중단이 발생하는 경우에 대한 대비책이 미미하다.

따라서 본 연구에서는 일시적인 결함 및 사람의 운영상 실수에 대처하여 중단없는 서비스를 제공할 수 있는 기술을 제안한다. 그리고 예기치 못한 결함이 발생하여 이를 해결하고자 하는 일련의 과정에서 사람의 개입을 최소화할 수 있도록 시스템 스스로 자원을 관리할 수 있는 기법을 제안한다. 또한 메모리 유출에 의한 결함을 자동으로 감지하고 관리할 수 있는 자원모델링을 통해 제안한 기법의 실효성을 살펴본다.

본 연구는 과학기술부 21세기프론티어연구개발사업의 일환으로 추진되고 있는 유비쿼터스컴퓨팅및네트워크원천기반기술개발사업의 지원에 의한 것임.
이 논문은 2004년도 두뇌한국21사업에 의하여 지원되었음.

2. 관련연구

IBM에서 오토노믹 컴퓨팅(autonomic computing)이라는 개념을 제창하였는데, 체온이 저하되면 열의 방출을 억제하기 위해 근육을 수축하고, 어두워 보이지 않을 경우 동공을 확장하는 것과 같이 뇌의 활동에 의한 명백한 사고없이 몸을 관리할 수 있는 자율신경계의 기능을 컴퓨터 시스템 관리에 적용하자는 것이다[2]. 오토노믹 컴퓨팅을 위한 제어 루프 구조는 크게 4부분(M.A.P.E.)으로 나눌 수 있는데, 자원의 상태 정보를 자동으로 수집하는 모니터링 부분(monitor), 수집된 정보를 근간으로 상황 분석을 수행하는 분석 부분(analyze), 분석된 자료를 바탕으로 사용자의 요구사항 및 서비스 수준 계약에 명시된 비기능적 요소를 만족하기 위한 정책을 계획하는 부분(plan), 수립된 계획에 준하여 적절한 조치를 수행하는 부분(execute)이다. 본 연구에서는 유비쿼터스 컴퓨팅 환경에 결함허용 요소를 첨가하기 위해 예기치 못한 결함

정보를 수집하고 수집된 정보를 분석하여 결함 발생 여부를 판단하고 결함이 발생하였을 경우 이를 치유하기 위한 도구를 선택하여 정책에 맞게 적용할 수 있는 기법을 제안하며, M.A.P.E. 부분으로 나눠 설명한다.

3. 모니터링 부분 (monitor part)

예측하지 못한 결함은 하드웨어적인 결함보다는 소프트웨어의 장시간 작동에 의해 발생하는 소프트웨어적인 결함에 의해서 발생하는데, 이를 소프트웨어 노화현상이라고 하며 예를 들면, 메모리 부족, 버퍼 오버플로우, 수학적 에러 누적 등이 있다[3]. 본 논문에서는 소프트웨어 노화현상을 일으키는 원인 중 메모리 유출에 의한 메모리 부족 현상으로 인한 결함에 대한 감지 및 복구 방안을 위한 자원모델링 기법을 설명한다. 또한 결함을 한 시스템 요소에서 발생할 수 있는 원인을 분석하여 해결하고자 하는 것이 아니며, 예기치 못한 결함이나 운영상 실수에 의한 결함을 시스템 자원의 전반적인 상태변화 감지를 통해 메모리 유출에 의해서 발생할 수 있는 결함 발생 상황으로 규정한다. 표 1은 메모리 유출을 검출하기 위해 자원의 상태를 감시해야 할 객체(object)와 수집한 정보의 의미(report)를 정리하였으며, 메모리 유출이 발생하였을 경우 해당 측정값의 변화(change)를 보여주고 있다. 즉, 결함이 발생하여 시스템에서 적절한 조치를 필요로 하는 상황이라 결론짓기 위해서 필요한 정보를 시스템 및 자원으로부터 선별·수집하는 것이다.

표 1 자원 객체와 메모리 유출시 변화

Object\Counter	Report	Change
Memory\Available Bytes	available bytes	fall
Memory\Committed Bytes	the private bytes committed to processes	rise
Process(process_name)\ Private Bytes	bytes allocated exclusively for a specific process	rise
Process(process_name)\ Working Set	the shared and private bytes allocated to a process	rise
Process(process_name)\ Page Faults/sec	the total number of faults (hard and soft faults) caused by a process	rise
Process(process_name)\ Page File Bytes	the size of the paging file	rise

4. 분석 부분 (analyze part)

수집된 정보를 근간으로 결함 발생 여부에 대해 분석하기 위해 결함 보고의 수, 유형, 심각성에 대해 테스트해야 되고 결함 보고 비율 및 밀도(단위 크기당 문제 보고 수)를 계산하여 결함에 대한 추세를 분석해야 한다. 이를 위해서는 객체지향 패러다임에서 사용하고 있는 구조적인 기술과 개념화 기술을 적용하여 필요로 하는 정보를 구성하고 이에 대한 해석을 수행하며, 결함 발생 근원지를 찾아낼 수 있어야 한다. 그림 1은 메모리 유출 현상에 의한 결함 발생 여부를 분석하기 위한 자원모델링으로 메모리의 사용가능 용량(Total byte available)이 한 프로세스가 실행되기 위해 필요한 메모리 용량(Minimum available)보다 작으면 결함이 발생할 수 있는 상황으로 분석한다.

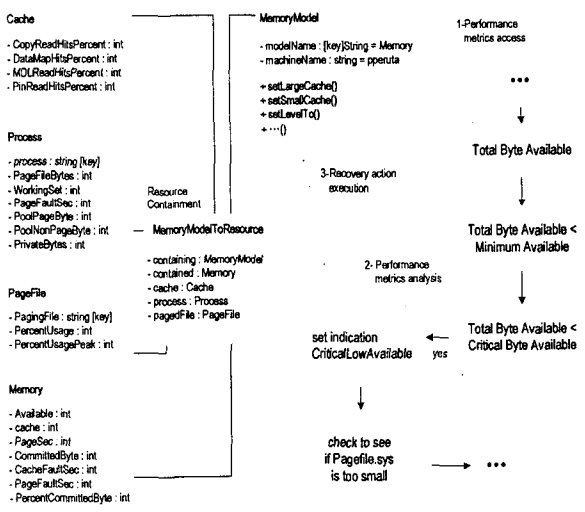

그림 1 메모리 유출 상황 분석을 위한 자원모델링

5. 계획 부분 (plan part)

시스템 구성을 위해 적용된 토폴로지는 유비쿼터스 컴퓨팅 환경에서는 시간 및 요구사항에 따라 변화할 수 있어야만 한다. 따라서 토폴로지에 사용된 컴포넌트(component, COM)와 각 컴포넌트에서 사용하고 있는 기반구조의 서비스(service, SVC)를 변화에 따라 적절히 매핑할 수 있어야만 한다. 이를 위해 이질적인 환경을 통합할 수 있는 미들웨어(예, OSGi, J2EE 등)에 대한 개발

이 활발히 진행되고 있는 것이다. 하지만 예기치 못한 결함이 매핑 과정에서 일어날 수 있으므로 새로운 컴포넌트를 개발하거나 기반구조의 서비스를 추가할 경우에는 기존 시스템에 미칠 수 있는 영향에 대한 분석을 수행해야 한다. 이를 위해서 전통적인 결함허용기법에서 사용하던 결함주입기법(fault injection technique)이 사용될 수 있으며, 이를 각 부분별로 표현하면 그림 2와 같다.

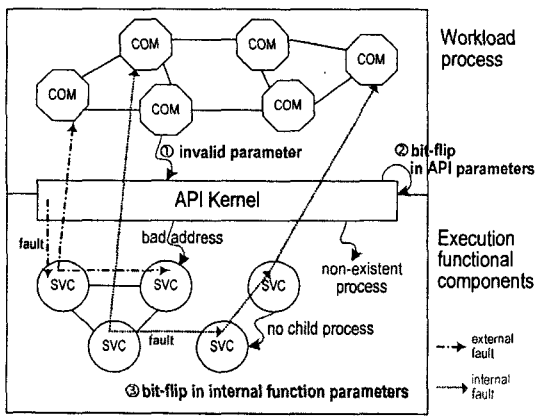

그림 2 컴포넌트 기반 시스템의 결함주입기법

결함주입기법에 의해서 한 컴포넌트 및 서비스에 결함이 발생하였을 경우 미칠 수 있는 영향에 대해 분석하고 이를 근간으로 결함이 발생하였을 경우 복구를 수행해야 되는 부분을 결정할 수 있는 정책을 마련한다. 더불어 각 서비스를 제공하기 위해 필요한 최소한의 자원에 대해서도 결정하는데, 홈 서버에서 보안을 위해 필요한 컴포넌트를 예로 결함에 대한 영향 및 최소자원요구량을 도식화하면 그림 3과 같다.

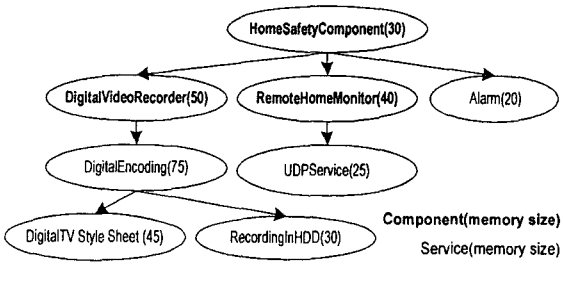

그림 3 결함 트리 및 최소자원요구량

6. 실행 부분 (execute part)

소프트웨어 노화현상에 의해서 결함이 발생하였을 경우 계획부분에서 선정된 정책에 따라 각 컴포넌트 및 서비스를 복구한다. 복구 기법은 본 논문의 사전 연구 결과 [4]로 개발된 소프트웨어 재활기법을 사용하는데, 소프트웨어 재활기법이란 향후 심각한 시스템 장애를 방지하기 위해서 시스템의 내부 상태를 청소(clean-up)하여 정상적인 상태에서 서비스가 가능하게 하는 프로액티브(pro-active)한 기법이다[4,5].

7. 결 론

사람, 컴퓨터와 사물이 통합된 환경에서 기능의 최적화를 이뤄야 하는 유비쿼터스 컴퓨팅을 실현하기 위해서는 사람의 개입을 최소화하면서 시스템 스스로가 관리할 수 있는 기술 개발이 필수적이며, 이를 위한 효율적인 대안이 오토노믹 컴퓨팅 개념이다.

따라서 본 논문에서는 오토노믹 컴퓨팅의 구조적 기본 개념인 M.A.P.E. 구조 설계를 근간으로 유비쿼터스 컴퓨팅 시스템의 예기치 못한 결함 발생에 대해 사람의 개입 없이 대처할 수 있는 결함허용기법을 제안하고 설계하였다. 향후에는 제안한 기법의 프로토타입 구현을 통해 검증을 수행할 예정이다.

참고 문헌

[1] M. Satyanarayanan, "Pervasive Computing: Vision and Challenges," IEEE Personal Communications, pp. 10-17, Aug. 2001.

[2] J. Kephart and D. Chess, "The Vision of Autonomic Computing," IEEE Computer, Vol. 36, No. 1, pp. 41-50, Jan. 2003.

[3] S. Garg, A. V. Moorsel, K. Vaidyanathan, and K. Trivedi, "A Methodology for Detection and Estimation of Software Aging," Proceedings of the 9th International Symposium on Software Reliability Engineering, pp. 282-292, Nov. 1998.

[4] C. Choi and S. Kim, "Self-configuring Algorithm for Software Fault Tolerance in (n, k)-way Cluster Systems," Lecture Notes in Computer Science, Springer, Vol. 2667, No. 1, pp. 742-751, May 2003.

[5] Y. Bao, X. Sun and K. Trivedi, "Adaptive Software Rejuvenation: Degradation Model and Rejuvenation Scheme," Proceedings of the International Conference on Dependable Systems and Network pp. 241-248, June 2003.