동적 모바일 생태계의 QoS 모니터 설계

이정목, 김문권, 장정란, 김수동
숭실대학교 컴퓨터학과
e-mail : jbluewing, mkdmkk , jrjang10, sdkim777@gmail.com

Design for QoS monitor of Dynamic Mobile Environment

Jung Mok Lee, Moon Kwon Kim, Jeong Ran Jang, Soo Dong Kim
Dept. of Computer Science, Soongsil University

요 약

모바일 컴퓨팅이 보편화 되고 기술이 발전 할수록, 모바일 환경에 가지고 있는 제약사항은 해결 해야 할 과제로 남아있다. 모바일 디바이스의 제약된 자원, 불안정한 네트워크, 서비스 접근성 등은 모바일 환경을 안정적으로 운영되기 힘든 요인이다. 이는 사용중인 어플리케이션의 서비스가 정지되거나 네트워크 전송의 실패 등의 문제가 빈번히 발생하는 원인이 된다. 본 연구에서는 이러한 문제 해결하기 위한 방법으로 정보를 효과적으로 수집하고 데이터를 분석하는 차세대 모바일 환경인 동적 모바일 생태계를 위한 모니터 시스템을 제시한다.

1. 서론

모바일 컴퓨팅은 CPU, 메모리, 네트워크 환경뿐만 아니라 높은 휴대성과 이동성을 가지고 있다.[1] 하지만 일반 데스크탑 컴퓨터나 서버용 컴퓨터에 비하여 성능이 떨어진다. 이는 모바일 컴퓨팅의 소형화로 자원들이 성능을 제약시켰기 때문이다. 모바일 어플리케이션의 일부 기능을 모바일 디바이스가 아닌 외부에 위치시키는 서버 기반 모바일 어플리케이션, 서비스 기반 모바일 어플리케이션에 대한 연구가 진행되고 있다[2][3]. 이런 연구 또한 모바일 컴퓨팅의 낮은 자원의 문제에 의해 어플리케이션들의 관리, 운영의 어려움을 가지고 있다. 그래서 본 논문은 자율적으로 서비스의 품질을 체크하고 서비스의 품질의 균형을 맞춰주는 동적 모바일 생태계 환경을 소개한다. 동적 모바일 생태계 환경을 구축하기 위해서는 자원의 모니터링, 품질측정, 서비스의 이동의 기능이 필요하다. 본 논문에서는 모바일 생태계 환경의 모니터링 및 품질 측정 기능 구현에 대하여 기술한다. 또한 모니터링을 위한 모바일 디바이스의 자원 사용량을 최소화하면서 필요한 데이터를 수집 하고, 시스템 관리자가 모니터를 통해 자원의 상태와 흐름을 파악할 수 있게 모니터 환경을 설계했다.

2 장에서는 관련연구, 3 장에서는 시스템의 소개와 모니터에 대한 설명을 하고 4 장에서는 모니터 설계에 관한 내용을 기술한다. 5 장에서는 4 장의 설계를 기반으로 구현 내용을 기술하고 6 장에서 결론을 내린다.

2. 관련연구

Liangzhao 의 연구에서는 웹서비스의 QoS 감지 미들웨어 AgFlow 를 제안하고 있다 [4]. 이 미들웨어는 웹서비스 혹은 복합 웹 서비스의 품질을 평가하여 좋은 품질의 웹서비스를 선택한다. 하지만 평가한 품질의 가시화와 웹서비스를 제공하고 있는 디바이스와 그 웹서비스를 사용하고 있는 디바이스에 대한 정보의 부재로, 평가된 품질에 대한 원인 분석이 어렵다.

Menasce, D.A.의 연구에서는 웹서비스의 QoS 를 측정 할 때 고려 해야 하는 항목에 대하여 제안하고 있다. 그 중 동시에 발생하는 서비스의 응답시간 을 측정 할 때는 평균뿐만 아니라 편차도 측정 해야 한다고 기술 하고 있다 [5].

La 의 연구에서는 서비스 제공 환경과 서비스 이용 환경이 안정적으로 운영될 수 있는 컴퓨팅 시스템 환경에인 자율 안정을 위한 서비스 기반 모바일 컴퓨팅 에코시스템을 연구하였다 [6]. 안정적인 시스템 환경을 유지하기 위한 품질 측정 방법과 시스템의 4 단계 프로세스로 자율 안정 시스템의 구조를 제안하고 실험하였다. 하지만 자율안정을 위한 시스템 모니터링에 대한 구체적으로 다루고 있지 않다.

3. DME 의 QoS 모니터의 기능

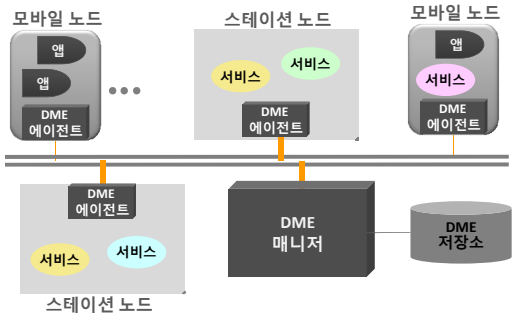
동적 모바일 생태계의 전반적인 설명과 품질 측정 방법과 모니터링 기법에 대하여 설명한다.

3.1. 동적 모바일 생태계

동적 모바일 생태계(Dynamic Mobile Ecosystem, DME)는 모바일 환경에서 모바일 어플리케이션과 서비스, 노드들이 일정한 수준의 품질을 유지하기 위해서 실시간으로 품질을 측정하고, 동적으로 전체 환경을 관리한다. DME 는 크게 생태계 모니터 기능, 서비스나 어플리케이션 동적 이동 기능이 있고, 그 구조는 ((그림 1)와 같다. 연구 대상인 모니터는 DME 매니저, DME 에이전트, 그리고 DME 저장소 3 요소로

이루어져있다.

DME 에이전트는 어플리케이션이나 서비스를 가지는 노드에 설치되며, 관리 대상 어플리케이션이나 서버의 정보 수집하고 디바이스의 자원 상태를 전송한다.



(그림 1) DME 의구조

DME 매니저는 DME 환경을 관리하며, DME 에이전트가 전송한 수집된 데이터를 바탕으로 DME 환경에 대한 품질을 모니터하고, 상태에 따라, 다른 노드로 서비스를 이동이나 복제를 명령을 하거나 서비스 경로를 변경하는 결정을 한다.

3.2. QoS 모니터의 기능

QoS 모니터의 기능은 각 디바이스의 자원을 수집하는 기능, 수집한 자원을 모니터에 표시하여 문제점을 빠르게 분석 할 수 있게 하는 기능, 분석된 문제점을 로그로 보여주는 기능이 있다.

디바이스 자원수집은 각 디바이스에 설치된 DME 에이전트에서 CPU 사용량, 메모리 사용량, 네트워크 대역폭, 배터리 량 등을 서버로 전송 하면 DME 저장소에 저장을 한다. 자원의 수집 시 각각 자원 별 수집시간을 다르게 하고 전송 시 가변 율을 두어 최대한 디바이스 자원을 적게 사용한다.

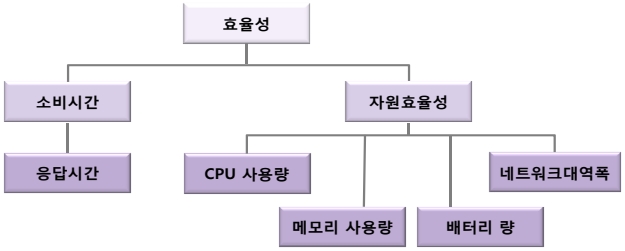
수집한 자원 디스플레이 기능은 멀티 차트와 동적 차트를 사용을 하여 한눈에 데이터의 흐름을 인지 하고 임계 점을 차트에 표시를 함으로서 문제점이 있는 부분을 빠르게 인지가 가능하도록 한다.

로그 기능은 생태계의 균형을 깨뜨리는 데이터 결과물을 화면에 표시한다.

3.3. DME 의 품질 모델

DME 매니저는 실시간으로 전반적인 DME 환경을 모니터링 하여 현재의 DME 상태를 DME 관리자에게 전달한다. DME 의 상태는 시스템의 품질을 의미함으로 시스템 품질을 측정하기 위한 품질 모델이 필요하다. 따라서 소프트웨어의 품질을 객관적이고 정량적으로 측정할 수 있고, 품질 모델의 표준으로 인정 받고 있는 ISO 9126[7]을 사용하여 모니터링 품질 모델을 결정한다.

DME 는 전체 시스템이 일정 이상의 성능을 유지하기 위한 시스템 운영 방식을 취하고 있기 때문에, 성능이 최우선적으로 고려되어야 한다. 본 연구에서는 성능과 밀접한 관련이 있는 효율성(Efficiency)을 품질 모델로 정하고, 소비시간(Time consumption)과 자원 효율성(Resource utilization) 속성을 정의한다.



(그림 2) DME 품질 모델과 속성

품질 속성은 ((그림 2)와 같이 소비시간의 속성으로 응답시간, 자원 효율성으로는 CPU 사용량, 메모리 사용량, 배터리 량, 네트워크 대역폭이 있다. 응답시간은 어플리케이션이 요구하는 서비스를 요청하는 시간부터 서비스 수행이 끝나고 결과를 어플리케이션으로 전송하는 시간까지를 의미이다. 각 속성들의 메트릭은 <표 1 과 같다.

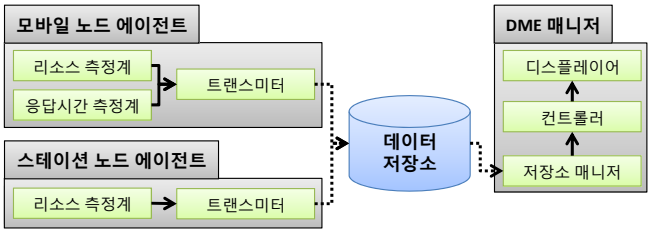
<표 1> 속성별 품질 메트릭

속성	메트릭
응답시간	$RT(app, service) =$ $network\ time + service\ computation\ time$
CPU 사용량	$CPUusage(node) =$ $current\ cpu\ usage / total\ cpu\ usage$
메모리 사용량	$Memoryusage(node) =$ $current\ memory\ usage / total\ memory\ usage$
배터리량	$Bettery(node) =$ $remain\ bettery / full\ bettery$
네트워크 대역폭	없음

4. QoS 모니터 설계

4.1. 아키텍처 설계

DME 모니터의 아키텍처는 크게 모바일 노드 에이전트, 스테이션 노드 에이전트, 매니저로 구성된다. (그림 3 은 이들을 구성하는 컴포넌트와 이들의 연관관계를 보여준다.



(그림 3). DME 아키텍처

모바일 노드 에이전트는 디바이스의 리소스 사용 상태를 측정하는 리소스 측정계, 서비스의 응답시간을 측정하는 응답시간 측정계, 측정된 데이터를 데이터저장소에 저장하는 트랜스미터로 구성된다.

스테이션 노드 에이전트는 리소스 측정계와 트랜스미터로 구성된다. 스테이션 노드는 서비스를 제공하는 역할을 하므로 응답시간 측정계는 가지지 않는다. DME **매니저**는 필요한 데이터를 데이터 저장소로부터

요청하여 획득한 후 재구성하는 저장소 매니저, 데이터를 획득, 재구성 하도록 저장소 매니저에 요청하고 이들 데이터를 화면에 제공하는 컨트롤러, 정보를 사용자에게 보여주는 화면으로 구성된다. 여러 모바일/스테이션 노드에 에이전트가 탑재되어 해당 모바일/스테이션의 상태를 모니터링하고 이들을 매니저에서 모니터링 한다.

4.2. QoS 모니터링 기법

QoS 모니터링 기법은 데이터 모니터링을 하고 모니터링 데이터의 QoS 을 측정하는 방법 순으로 진행된다.

• 데이터 모니터링 방법

데이터를 모니터링 하는 에이전트는 주기적으로 서비스 품질을 보장하기 위한 데이터를 풀링 모델[8] 방식으로 수집한다. 에이전트에 수집된 데이터 관리 시스템으로 전송하게 된다. 하지만 모니터링 결과를 정기적으로 전송하는 일반적인 방식은 중복된 데이터까지도 전송함으로써, 모바일 디바이스의 전송에 드는 비용 즉, 배터리와 네트워크 자원 등을 소비하게 하는 결과를 초래한다. 따라서 모바일 디바이스를 위한 모니터링 전송방법을 다음과 같이 제시한다.

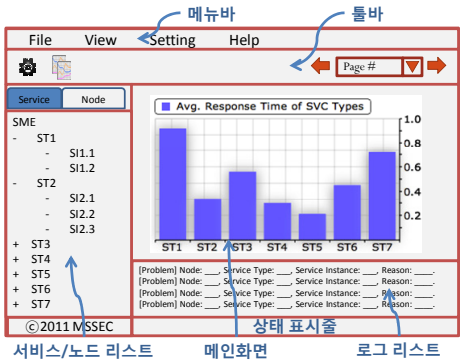
에이전트는 변수와 같이 임시 데이터 저장소가 필요하다. 임시 데이터 저장소는 최근 전송된 데이터를 저장하고 있다. 자원 소비량을 최소화하기 위해서 에이전트가 데이터를 전송하기 전에, 임시 저장소의 데이터와 비교하여 일정 범위 이상의 차이가 있을 경우에만, 관리 시스템으로 데이터 전송하여 불필요한 낭비를 사전에 방지한다.

• 모니터링 데이터의 QoS 측정 방법

품질은 어플리케이션 제공자와 서비스 제공자에 의해 작성된 서비스 수준 계약서 (Service Level Assignment, SLA) 를 통해 상태를 확인한다. SLA 에는 품질 속성들의 최저임계치인 Threshold_low(x)과 최고임계치인 Threshold_hgih(x)을 설정하여 현재 데이터와 비교하여 품질을 평가한다.

4.3. 사용자 인터페이스 설계

DME 의 사용자 인터페이스는 단순성, 편의성, 유용성의 설계 원칙을 따르며 이를 (그림 4) 에서 보여준다.



(그림 4) DME 의 사용자 인터페이스 설계

단순성이란 메뉴와 화면의 구성이 복잡하지 않아 쉽게 관련 기능을 수행 할 수 있고 정보 전달이 분명함을 의미한다. 편의성이란 사용자가 쉽게 기능을 수행하고 정보를 획득할 수 있음을 의미하며 단순성에서 파생된다. 유용성이란 사용자가 필요로 하는 정보를 정확하고 효율적으로 전달함을 의미한다..

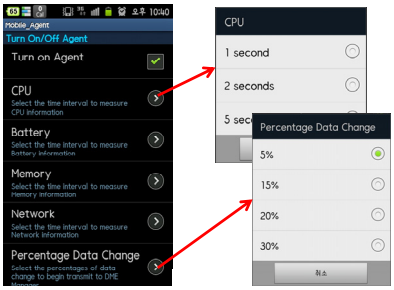
화면은 크게 메뉴바, 툴바, 서비스/노드 리스트, 메인화면, 로그 리스트로 나누어 진다. 메뉴바에서는 사용자가 모든 기능과 설정을 수행 할 수 있도록 한다. 툴바에서는 모든 메뉴 중 자주 사용하거나 중요한 기능을 위치시켜 사용자에게 편의성을 제공한다. 서비스/노드 리스트에서는 모니터링 대상인 모든 서비스와 모바일/스테이션 노드를 트리 구조로 보여주고 사용자가 한 항목을 선택하면 해당 정보를 메인 화면에 라인 차트, 바 차트, 표와 같은 형태로 보여준다. 로그 리스트는 모니터링 중에 문제점이 발생하면 그 정보를 사용자에게 전달하기 위해 존재한다.

5. 구현 및 실험

구현은 디바이스의 자원상태를 수집하고 전송하는 DME 에이전트 구현과 수집된 디바이스들의 자원 정보를 모니터링 하는 DME 매니저 모니터링으로 나누어져 있다.

5.1. Agent 구현

수집할 데이터 각각의 수집 시간과 전송 가변 율은 ((그림 5)의 화면에서 사용자가 입력을 할 수 있도록 한다.



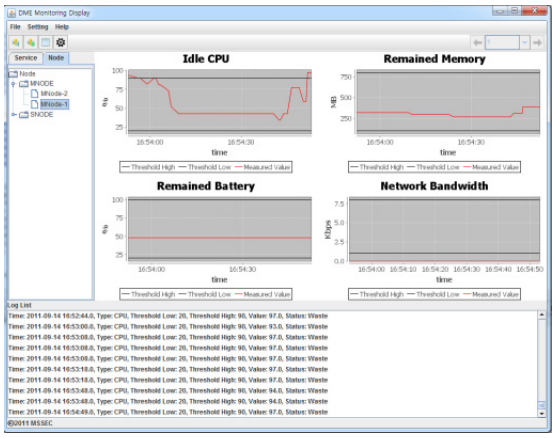
(그림 5)사용자 인터페이스 구현

모바일은 4 종류의 데이터를 수집한다. CPU 의 상태는 안드로이드 OS 의 리눅스 명령어를 실행하여 결과값을 계산하는 방법을 채택하고 배터리와 메모리 상태는 안드로이드 API 에서 제공하는 클래스를 이용하여 자원을 수집하였다. 네트워크 속도 측정은 항상 열려있는 안정적인 서버에 파일을 전송해서 시간을 측정 하는 방법을 채택하였다.

각각의 데이터의 전송은 JSON 포맷으로 안드로이드의 HTTPClient 클래스를 사용하여 SME 저장소로 데이터를 전송한다.

5.2. Manager 모니터링 구현

멀티 차트에서 각 화면에 보여주는 데이터는 시간별로 평균화 시켜서 JFreeChart 의 동적 차트에 보여준다. 화면은 크게 서비스와 노드 별로 나누어진다. 서비스는 해당 서비스의 응답시간을 중점적으로 보여주고, 노드는 해당 노드의 사용 가능한 CPU, 메모리, 남은 배터리, 네트워크 속도를 화면에 보여준다. ((그림 6)은 매니저의 노드 화면을 보여주고 있다.

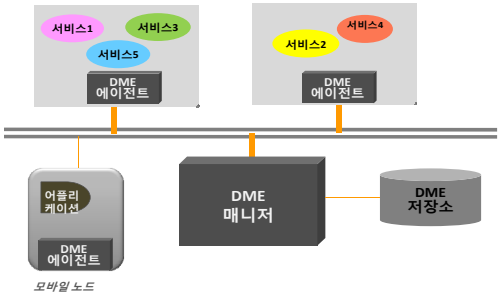


(그림 6) DME Manager 구현 화면

그래프에 사용자가 설정한 임계치가 검은 선으로 나타나고 붉은 선으로 해당 자원의 상태를 나타낸다. 임계치를 넘은 자원은 로그 리스트에 해당 자원의 로그를 남긴다

5.3. 실험 환경 및 유형

DME 의 실험은 ((그림 7)과 같이 안드로이드 플랫폼이 탑재된 모바일 디바이스, 서로 다른 5 개의 서비스가 배포된 서버 2 대, 데이터를 저장/관리 하는 저장소, 모든 정보를 총괄하는 DME 매니저가 있다.



(그림 7) 실험 환경

본 실험에서는 어플리케이션이 서비스 1, 서비스 2, 서비스 3, 서비스 4, 서비스 5 를 병렬로 호출 한다.

서비스 1 은 0.02 초의 실행시간이 걸리고, 서비스 2 는 0.04 초, 서비스 3 는 0.06 초, 서비스 4 는 0.08 초, 서비스 5 는 0.1 초의 실행 시간이 걸린다. 서비스 응답시간의 임계치는 최하 0.05 최고 0.1 로 설정한다.

5.4. 실험 결과 및 분석

어플리케이션을 실행하면 서비스 1 은 0.02 초후, 서비스 2 은 0.04 초후, 서비스 3 은 0.06 초후, 서비스 4 은 0.08 초후, 서비스 5 는 0.1 초후에 해당 서비스 디스플레이 화면 그래프가 변화 되는 것을 확인 할 수 있고 서비스 1, 서비스 2 에 해당하는 서비스 응답시간이 설정한 임계치를 벗어 났기 때문에 로그 리스트에 로그를 남긴다. 그리고 화면의 노드 의 상태는 어플리케이션이 실행 될 때 CPU 와 메모리의 상태가 변화 하는 것을 확인 하였다.Memory, CPU, 네트워크 대역폭, 배터리 같은 자원은 지정한 가변율을 벗어 나지 않으면 해당 자원의 데이터를 보내지 않는 것 또한 확인 가능하였다.

메모리, CPU, 네트워크 대역폭, 배터리 등의 자원의 상태가 임계치를 벗어 나는 지와 서비스의 응답 시간이 임계치를 벗어 나는 지를 쉽게 확인가능 함으로서 자원의 품질측정 및 모니터링 기능을 효과적으로 수행 하는 것을 확인 할 수 있다. 또한 DME 에이전트에서 자원의 가변율에 따라 데이터를 전송 함으로서 모바일 디바이스의 네트워크 자원과 배터리 자원을 좀더 효율적으로 사용 할 수 있게 되었다.

6. 결론

본 논문에서는 서비스 품질을 위해 동적으로 서비스 품질을 체크하여 균형을 맞춰주는 DME 시스템을 소개하였다. 그리고 DME 시스템의 QoS 모니터 설계를 보여주었다. 또한 실험환경을 구축하여 설계에 따라 구현한 모니터로 실험을 진행하여 그에 대한 결과를 분석하여 본 논문에서 다루는 연구 결과의 실효성을 보여주었다.

Acknowledgement

이 논문은 정보통신산업진흥원의 SW 공학 요소기술 연구개발사업에 의해 지원되었음을 밝힙니다.

이 논문은 2009 년 정부(교육과학기술부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구입니다. (No. 2009-0076392)

참고문헌

[1] König-Ries, B. and Jena, F., “Challenges in Mobile Application Development,” *it-Information Technology*, Vol. 52, No. 2, pp. 69-71, 2009.

[2] Natchetoi, Y., Kaufman, V., and Shapiro, A., "Service-oriented architecture for mobile applications," *In Proceedings of the 1st international workshop on Software architectures and mobility (SAM '08),* pp.27-32, 2008.

[3] Tergujeff, R., Haajanen, Jyrki, Juha, J., and Toivonen, L.S., “Mobile SOA: Service Orientation on Lightweight Mobile Devices,” *In Proceeding of 2007 IEEE International Conference on Web Services (ICWS 2007)*, pp. 1224-1225, 2007.

[4] Liangzhao Zeng, Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., and Chang, H., “QoS-aware middleware for Web services composition,” *IEEE Transactions on Software Engineering*, Vol. 30, No. 5, pp.311-327, 2004.

[5] Menasce, D.A, “Qos issues in Web services,” *IEEE Internet Computing*, vol. 6, no. 6, pp. 72 - 75, Dec 2002.

[6] Hyun Jung La, Jeong Ran Jang and Soo Dong Kim, "Self-Stabilizing Ecosystem for Service-based Mobile Computing ," *To appear In Proceedings of the 8th IEEE International Conference on e-Business Engineering (ICEBE 2011).*

[7] ISO/IEC, ISO-IEC 9126-1 Software Engineering – Product Quality – Part 1: Quality Model, 2001.

[8] Du Wan Cheun, Hyun Min Lee, and Soo Dong Kim, "An Effective Framework for Monitoring Service-based Mobile Applications ," *In Proceedings of the 7th IEEE International Conference on e-Business Engineering (ICEBE 2010)*, pp. 278-283, 2010.