

유비쿼터스 환경을 위한

위치 감지 프로세스 전략의 자동화 구조

최호영⁰ 최창열 김성수

아주대학교 정보통신전문대학원 정보통신공학과

{funkyleo⁰, cicho, sskim}@ajou.ac.kr

An Autonomic Architecture of Location Detection Process Strategies for Ubiquitous Environment

Hoyoung Choi⁰ Changyeol Choi Sungsoo Kim

Graduate School of Information and Communication, Ajou University

요 약

유비쿼터스 환경에서 이동하는 객체 위치를 감지하고 서비스를 제공하는 위치 감지 시스템의 역할은 매우 중요하다. 하지만, 감지 시스템들은 적용되는 감지 범위에 따라 프로세스 전략이 다르고 사용이 제한적이기 때문에 신뢰성 있는 QoS를 보장하기가 힘든 실정이다. 따라서 본 논문에서는 지능적인 모니터링 기법을 사용하여 상황을 분석하고 자동으로 최적의 감지 프로세스 전략을 선택하는 자동화 시스템을 설계하였다. 또한 네트워크 대역폭 변화에 따라 최적 QoS를 보장할 수 있는 제한된 시스템의 성능을 시뮬레이션을 통해 검증한다.

1. 서 론

최근 모든 기술 분야에 유비쿼터스 컴퓨팅 개념이 도입되고 있다[1]. 유비쿼터스 컴퓨팅이란 언제, 어디서나 사용자가 신뢰성 높은 서비스를 지속적으로 받을 수 있는 환경이다. 특히, 위치 감지 시스템(Location Detection System)을 통해 이동 객체의 위치를 파악하고 사용자가 요구하는 서비스를 정확히 제공한다[1]. 따라서 효율적인 위치 감지 기술을 적용하여 유비쿼터스 환경에 존재하는 사용자들에게 높은 QoS(Quality of Service)를 제공하는 것이 중요하다. 최근 활발하게 연구되고 있는 Wi-Fi와 GPS(Global Positioning System)가 대표적인 위치 감지 기술들이다. 그런데, 이러한 기술들은 감지 대상 범위가 조금씩 다르고, 설계될 때에도 상이한 모델들을 이용하기 때문에 프로세스 전략이 저마다 틀리며, 그 결과 위치 감지 시스템의 QoS를 높이는 방법이 제한적이다. 이런 문제점은 적용모델 매핑(Mapping)을 통하여 네트워크 대역폭 등 자원 상태에 따라 적합한 프로세스 전략을 사용함으로써 해결할 수 있지만[2], 능동적인 프로세스 선택을 도와줄 수 있는 모니터링 기법과 변화하는 사용자 상황에 따라 자율적으로 프로세스 전략을 선택하는 기술이 마련되지 않으면 이는 불가능하다. 따라서 본 논문의 목적은 위와 같은 문제점을 해결하기 위한 것으로, 양질의 위치

기반 서비스를 제공할 수 있는 상황 적응 프로세스 제어기와 오버헤드없이 자율적으로 데이터들을 수집하는 자율 모니터링 관리자를 제안한다. 특히, 시스템 구조 설계는 자율 컴퓨팅 이론[3]에 의거하여, 이동 객체 감지 프로세스 전략을 자동화하고 운영자의 인위적인 관리없이 시스템 스스로 주어진 환경에서 최적 QoS를 보장할 수 있도록 한다. 그리고 설계된 구조는 시뮬레이션을 통하여 성능 측정 및 검증한다.

2. 관련 연구

유비쿼터스 환경에서 이동 객체 위치를 감지하는 시스템들은 크게 두 가지 모델로 설계될 수 있다[4]. 하나는 발생 이벤트들을 수집하기 위해 정해진 기간 동안 시스템을 점검하는 시한 모델(Timed model)로서 폴(Poll) 프로세스 전략 방식을 사용한다. 폴 전략은 폴 간격(Poll Interval) 동안 발생한 위치 변화 이벤트들을 수집한다. 다른 하나는 이동 객체가 위치를 변화시킬 때마다 이벤트를 발생시켜 그것을 처리하도록 하는 동기 모델(Synchronous model)로서 드롭(Drop)과 연기드롭(Defer & Drop) 프로세스 전략 방식을 사용한다. 드롭 전략은 발생 이벤트들이 정해진 시간 간격(Time Slice)에 모두 전송되지 못할 경우에는 초과된 이벤트들을 탈락시킨다. 연기드롭 전략에서는 정해진 시간 간격에 전송 가능한 데이터 양을 초과한 나머지 이벤트들은 일정 시간동안 전송이 연기(Defer)되고 연기 시간(Defer Time)마저 초과하게 되면

본 연구는 과학기술부 21세기 프론티어 연구 개발 사업의 일환으로 추진되고 있는 유비쿼터스 컴퓨팅 및 네트워크 원천기반 기술개발사업의 지원에 의한 것임.

이 논문은 2004년도 두뇌한국21사업에 의하여 지원되었음.

탈락(drop)된다. 위치 검색 시스템(Locating System)은 시한 모델에 속하고, 위치 식별 시스템(Location System)은 동기 모델에 속한다. GPS와 같은 위치 검색 시스템은 이동 반경이 넓은 객체를 검색하는데 있어 높은 성능을 나타내지만, 빌딩과 같이 제한된 환경에서는 객체 식별이 낮은 효율성을 나타낸다. 반면, Wi-Fi와 같은 위치 식별 시스템은 이동 범위가 큰 객체 위치를 검색하는 것에서는 위치 검색 시스템보다 성능이 떨어지는 단점을 가지지만, 특정 범위 내에서 식별된 객체에 대해 정확하고 다양한 서비스를 제공할 수 있다. 동기 모델과 시한 모델은 위치와 객체를 매핑함으로써 각 시스템에 대해 상호 프로세스 전략을 적용할 수 있다[5]. 본 논문에서 제안된 구조 및 매커니즘의 응용분야는 이 두 모델을 매핑한 위치 감지 영역이다.

3. 자동화 관리자

복잡한 시스템 환경에서 효율성과 생산성을 높이기 위해 시스템 스스로가 주어진 문제에 대해 자율적으로 반응하고 대처하는 자율 컴퓨팅이 최근 이슈가 되고 있다. 자율 컴퓨팅 핵심을 이루는 자동화 관리자(Autonomic Manager)는 역할에 따라 4가지로 분류된다. 대상 요소 데이터들을 수집하고, 분류하여 가공하는 모니터링 부분과 시스템 상황을 파악하고 해결 방안을 제시하는 분석 부분이 있다. 그리고 문제 해결 목표를 이루기 위해 계획과 방안을 설계하는 계획 부분과 시스템 동작 중 계획된 내용을 적용하기 위해 필요한 영역을 제어하는 실행 부분이 있다. 본 연구에서는 모니터링 부분과 분석 부분의 자동화를 목표로 시스템을 설계한다.

4. 자동화 시스템 기본 구조

그림 1은 제안된 자동화 시스템을 보여주며 구조는 크게 자율 모니터링 관리자(Self-Monitoring Manager)와 적응 프로세스 제어기(Adaptive Process Controller)로 구분된다. 자율 모니터링 관리자는 위치 시스템에서 서비스를 요구하는 객체 정보를 모니터링하고 적응 프로세스 제어기로 전송하는 기능을 수행한다.

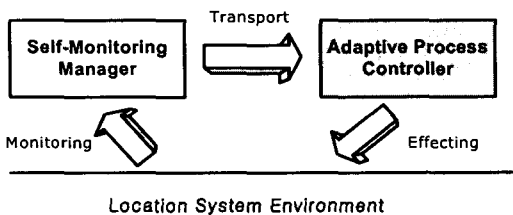


그림 1. 자동화 구조의 개념도

적응 프로세스 제어기는 그 내용을 기반으로 프로세스 전략을 선택하고 위치 감지 시스템에 적용시킨다. 적용시킨 후 모니터링 관리자를 통해 결과를 다시 관찰한다. 이처럼 피드백을 통해 주어진 상황에 맞는 최적 프로세스를 선택하기 때문에 시스템 QoS를 더욱 향상시킬 수 있다.

4.1 자율 모니터링 관리자

자율 모니터링 관리자는 오버헤드없이 네트워크 대역폭과 시스템 QoS를 관찰하고 분석하여 프로세스 전략 적용 여부를 판단하고 해당 데이터들을 적응 프로세스 제어기로 전송하는 기능을 담당한다. 자율 모니터링 관리자는 그림 2에서 보는 바와 같이 탐지기(Probe)와 로컬 버퍼로 구성된다.

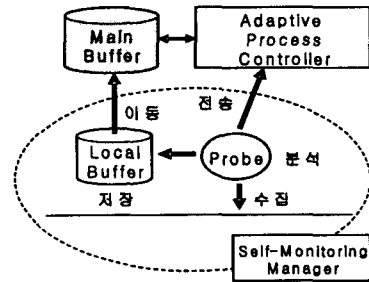


그림 2. 자율 모니터링 매니저 구조도

탐지기는 네트워크 대역폭과 시스템 QoS 데이터를 수집하여 실시간으로 분석함으로써 현재 성능을 관찰하는 기능을 가지며 로컬 버퍼는 탐지기에 의해 수집된 데이터들을 임시적으로 보관하는 기능을 가진다. 이유는 전송으로 생기는 오버헤드를 최소화하기 위해서이며 수집된 데이터가 성능에 위배되지 않는 정보라면 적응 프로세스 제어기에 전송하지 않고 로컬 버퍼에 저장한다. 그리고 버퍼 용량이 다하면 저장된 데이터들을 메인 버퍼로 이동시킨다(그림 3 참조).

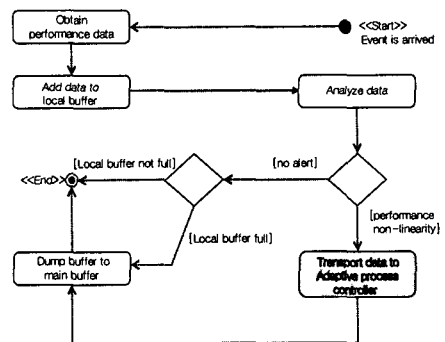


그림 3. UML 활동 다이어그램

4.2 적응 프로세스 제어기

자율 모니터링 매니저에서 전송된 데이터들은 적응 프로세스 제어기로 입력된다. 감지 프로세스 전략 테이블을 이용하여 전송된 데이터들은 시스템 자원 상태에 맞는 프로세스 전략을 선택하는데 사용된다. 감지 프로세스 전략 테이블은 기존 QoS 관련 연구 자료 [5]들을 분석하여 작성한 것이다. 연구 자료에 따르면 대역폭이 32Mbps 이하일 경우, 폴 전략은 다른 전략보다 높은 QoS를 보장하고 대역폭이 71Mbps 이상일 경우에는 연가·드롭(DD) 전략이 가장 높은 QoS를 보장하는 것으로 들어났다. 그림 5는 프로세스 전략 선택 알고리즘으로써 감지 프로세스 전략 테이블 정책에 따라 전략들을 적용시킨다. 전략들의 정상적인 동작 확인을 위해 현재 대역폭에서 보장할 수 있는 최고의 QoS를 한계범위(QoS Boundary)로 제한한다. 그리고 차선 프로세스 전략을 두어 주 전략이 오동작을 일으킬 경우, 대신 작동하여 신뢰성있는 알고리즘을 실현할 수 있도록 돕는다.

QoS Boundary	Process Strategies	Bandwidth
Below 43 %	1) Poll 2) DD 3) Drop	Until 32
Between 43 - 68 %	1) DD 2) Poll 3) Drop	Until 71
Above 68 %	1) DD 2) Drop 3) Poll	Over 71

그림 4. 감지 프로세스 전략 테이블 (좌)

그림 5. 프로세스 전략 선택 알고리즘 (우)

```

Data arrival
If (QoS Boundary < 43) {
  If (Bandwidth ≤ 32)
    Adapt_Strategy (Poll)
  Else
    Adapt_Strategy (DD)
}
Else {
  If (QoS Boundary > 68) {
    If (Bandwidth > 71)
      Adapt_Strategy (DD)
    Else
      Adapt_Strategy (Drop)
  }
  Else {
    If (32 < Bandwidth ≤ 71)
      Adapt_Strategy (DD)
    Else
      Adapt_Strategy (Poll)
  }
}
    
```

5. 성능 평가

제안된 시스템 성능을 측정하기 위해 [5]에서 사용한 파라미터를 기준으로 시뮬레이션을 수행했다. 사용자의 수는 10⁴로 지정하고 10² 시간 사이클 동안 시뮬레이션을 동작시켰다. 시스템의 최대 네트워크 대역폭은 140 Mbps로 제한했다. 폴 전략 QoS는 발생한 모든 위치 변화에 대해 성공적으로 감지된 이벤트 수로 정의하였고, 드롭과 연가·드롭의 QoS는 모든 발생 이벤트에 대해 성공적으로 전송된 이벤트 수로 정의했다. 그림 6에서는 시간과 대역폭에 따라 변화하는 제안된 시스템의 QoS를 보여준다. 낮은 대역폭에서 위치 시스템 QoS는 낮게 표시되지만, 대역폭 40Mbps부터는 원만한 QoS 향상이 이루어지는 것을 확인할 수 있다.

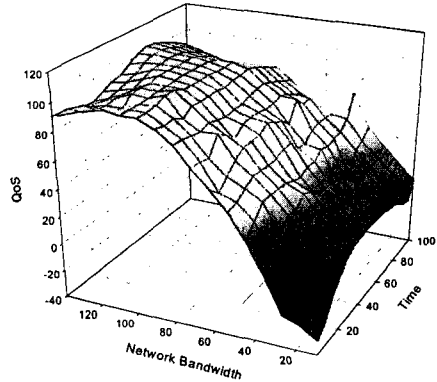


그림 6. 대역폭에 따른 QoS 변화

6. 결론

유비쿼터스 환경에서 위치 감지 시스템은 이동 객체의 위치를 감지하고 요구 사항을 파악하는 중요한 요소이기 때문에, QoS 보장 기법 마련이 시급하다. 본 논문에서 제안된 프로세스 전략 자동화 구조는 위치 시스템 대역폭과 QoS를 관찰하여 자율적으로 최적 프로세스 전략을 선택함으로써 사용자에게 높은 QoS를 보장할 수 있었다. 제안된 구조에서는 관리자 모니터링 부분과 분석 부분에서 자동화를 이루었지만 향후 시스템 상황을 예측하고 자율적으로 프로세스 전략 테이블을 진화시키는 기법을 연구하여 관리자 계획 부분에서도 자동화를 이룰 계획이다.

참고문헌

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

[2] S. Fischmeister and G. Menkhau, "L2: A Novel Concept for Cell-based Location-Aware Services," Salzburg University Technical Report C45, 2002.

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

[4] C. Kirsch, "Principles of Real-Time Programming," Proceedings of Second International Conference on Embedded Software, pp. 61-75, Oct. 2002.

[5] S. Fischmeister, G. Menkhau, and A. Stumpf, "Location-Detection Strategies in Pervasive Computing Environments," Proceedings of IEEE Pervasive Computing, pp. 23-26, Mar. 2003.