유연한 응답 기능을 가지는 이동 에이전트에 기반을 둔 공급 체인 관리

정 원 호† · 남 희 정††

요 약

이동 에이전트는 분산 응용에 있어서 네트워크의 부하와 대기시간을 줄일 수 있는 좋은 방법들 중 하나이며, 클라이언트/서버, 애플릿/서블릿, Code-on-Demand 등의 개념을 포괄하는 새로운 소프트웨어 패러다임이라 할 수 있다. 본 논문에서는 다양한 분산 응용에 있어서 결과물 응답 시점에서 고려될 수 있는 여러 상황에 대해 유연하게 대처할 수 있는 응답 기능을 가진 이동 에이전트 개념이 제안된다. 결과물 응답 시점에서 고려될 수 있는 상황으로 여러 가지가 있을 수 있는데, 1) 부담스러운 결과물의 크기, 2) 작업 유형에 따른 결과물에 대한 다른 방식의 응답 필요, 3) 결과물 응답 대상 노드로의 접속 불가능 상황, 그리고 4) 여정을 맞추기 위한 작업 에이전트의 신속한 이동의 필요, 등이 있을 수 있으며, 다른 한편으로, 소스 노드 입장에서는 원격 작업에 대한 수행 여부의 파악이 있을 수 있다. 이러한 상황에 유연하게 대처할 수 있는 3가지 응답 방식이 제안되며, 제안된 응답 방식은 응답 대상 노드들이 우선순위로 지정된 우선순위 응답 리스트를 사용하고 있다. 그리고, 이에 기반을 둔 이동 에이전트를 사용하여 간단한 공급 체인 관리 실험 모델이 설계 구현된다. 이는 공급 체인에서 공급자들로부터 적시에 작업을 수행하고 정보를 얻어내는 것이 중요하며, 또한 협동 작업 같은 다양한 작업의 형태가 존재하므로 결과물을 경우에 따라 다르게 응답해야 할 필요가 있어 제안된 응답 기능을 사용하기에 적합한 좋은 분산 응용 중의 하나이기 때문이다.

A Supply Chain Management based on Mobile Agents with Flexible Reply Scheme

Won-Ho Chung† · Hee-Jung Nam††

ABSTRACT

Mobile agent is one of the promising ways of overcoming network load and latency. It is also a new software paradigm including those concepts of client/server, applet/servelet, and code-on-demand. In this paper, a new mobile agent concept with flexible reply scheme is proposed, which can deal with embarrassing situations when replying results should be accomplished in various distributed applications. For example, they are 1) a burden of bulky result, 2) a need of different reply scheme dependent on work type, 3) connection failure to the target node, and 4) a need of fast migration to next node to keep the itinerary. Regarding the source node, there may be another situation that it wants to be aware of whether its work is completed or not. Three kinds of reply schemes are proposed for dealing flexibly with such situations. They are based on priority reply list where nodes to be replied are stored according to their priorities. An experimental supply chain management model using the proposed reply schemes is designed and implemented. It is one of good distributed applications appropriate for our reply schemes, because it requires different reply schemes according to work types and it is important to gather required information in time.

키워드 : 고정응답(Fixed reply), 공급체인관리(SCM), 다중응답(Multiple), 동적응답(dynamic reply), 이동 에이전트(mobile agent), 유연응답(flexible replay)

1. 서 론

분산 시스템 및 인공지능 분야에서 에이전트 개념이 등장하여 보편화되어 적용 되고 있고, 더 나아가 이를 확장시킨 다중 에이전트(multi-agent) 기법이 다양한 분야에서 광범위하게 연구되고 있다[1-5]. 그러나, 기존의 에이전트 개념은 비이동성 개념으로 네트워크 부하가 클 경우 혹은 접속 단절 상황에서 지연이 크고 작업 수행이 불가능하다는 문제점을 내포하고 있다. 이를 극복하고자 제안된 개념이 기존의 에이전트 개념에 이동 코드 기술을 접목시킨 이동 에이전트 기술이다. 그러므로, 이동 에이전트 기술은 기존의 클라이언트/서버, 애플릿/서블릿, 그리고 code-on-demand 개념을 통합하는 새로운 소프트웨어 패러다임이라 할 수 있으며, 구현하기에 따라 이동 에이전트가 가지는 특성은 매우 다양하다고 할 수 있다[6]. 그리하여, 최근에는 다양한 형태의 분산 정보 관리[7], 병렬 및 분산처리[8-9], 정보의 전송

※ 본 연구는 2000학년도 덕성여자대학교 연구비 지원으로 이루어졌음.
※ 본 연구는 KISTEP 여자대학교연구기반 확충사업의 연구비 지원으로 이루어졌음(00-N6-05-01-A-03).
† 정 회 원 : 덕성여자대학교 컴퓨터과학부 교수
†† 준 회 원 : 덕성여자대학교 대학원 전산및정보통신학과
논문접수 : 2001년 5월 11일, 심사완료 : 2001년 7월 3일

및 공유[10-11] 등, 그 응용 범위를 넓혀가고 있으며, 최근에는 SCM(Supply Chain Management)과 같은 전자상거래 분야에까지 이동 에이전트 적용을 위한 연구와 개발이 활발히 이루어지고 있다[12-18]. 이동 에이전트는 일단 목적하는 노드로 이동이 이루어지고 나면, 결과물 응답 전 까지 대부분의 작업이 해당 노드에서 로컬하게 수행되므로, 네트워크의 비접속 상태에서도 작업 수행이 가능하며, 작업 수행이 완료된 후 접속을 시도하여 결과물 응답을 하게된다. 그러므로, 이동 에이전트는 접속이 불안정하거나 부하가 높은 네트워크 환경에서 트래픽을 줄이고 채널을 효율적으로 사용할 수 있는 기대되는 해결책 중의 하나라고 할 수 있다. 이를 위해서는 에이전트에게 주어진 여정(itinerary)에 따라 효율적으로 이동이 이루어질 수 있도록 하는 동적 라우팅(dynamic routing)은 필수적이라 할 수 있으며, 많은 에이전트 시스템 또한 이를 지원하고 있다[6][19]. 동일한 맥락에서, 네트워크 분할(partition) 혹은 소스 노드의 크래쉬 등으로 에이전트의 작업 수행의 여부를 파악하기 힘든 상황에서 동일 작업의 중복 수행 문제가 발생할 수 있는데, Baumann 등은 이러한 중복 수행 문제를 줄일 수 있는 플랫폼인 Mole을 개발하였으며[19], 중복 수행 가능성을 줄이기 위해서 기존의 voting 기법을 기반으로 하는 프로토콜을 사용하고 있다[20]. 그러나, 여러 노드들이 특정 대상 노드를 모니터링하며 문제 발생시 그 역할을 다른 노드가 인수하는 등 그 과정이 복잡하다고 할 수 있다. 또한, 상대적으로 규모가 큰 에이전트인 경우 이동 전에 수행에 필요한 자원의 확보 여부를 조사한 후 에이전트를 이동 시키는 것이 네트워크 사용 측면에서 효율적이라 할 수 있다. 이를 위해, Bhandi는 규모가 큰 작업 에이전트를 직접 보내기 전에, 경량(lightweight)의 에이전트를 먼저 보내, 필요한 자원에 대한 확보 가능성을 확인한 후, 작업 에이전트를 보내어, 네트워크 채널의 효율적 이용을 꾀하고자 하는 기법을 제안하고 있다[21]. 이러한 연구들은 에이전트가 작업을 마치고 여정에 따라 다른 노드로 이동을 시작할 수 있다는 전제 하에서 효율적인 이동과 수행 여부에 관한 파악을 목적으로 하고 있다. 그러나, 에이전트가 다른 노드로 이동하기 위해서는 먼저, 현재 노드에서의 작업 수행에 대한 결과물의 응답이 응답 대상 노드로 이루어지고, 만일 결과물이 없는 경우라면, 작업 수행의 성공 여부에 관해서 응답이 이루어져야 하는 것이 일반적이다. 이러한 측면에서 볼 때, 결과물 응답은 여정에 따른 신속한 이동을 위한 선결 문제로써 혹은 작업 수행 사실에 대한 증거로써, 중요한 의미를 가지게 된다. 또 다른 측면에서 볼 때, Bhandi의 경량 에이전트 개념을 작업 결과물 응답에도 적용할 수 있으며, 네트워크 사용 측면에서 효과적이라 할 수 있다. 왜냐하면, 작업 수행 후 얻어지는 결과물이 부담스러운 규모일 수도 있기 때문이며, PDA와 같은 무선 저속 망의 경우에는 더욱 그러하다. 그리고, 에이전트의 이동 시점에서의 고려 사항과 동일한 맥락에서 결과물 응답을 적용해 보자. 만일, 결과물 응답 시점에서 네트워크 분할 혹은 특정 응답 대상 노드의 크래쉬로 인해 접속 시도가 실패한다면, 결과물 응답이 이루어질 수 없어 필요한 정보의 습득이 어려워지고, 또한 작업 수행 파악이 이루어지지 않아 전체 작업을 지연시킬 수도 있다. 더 나아가서는 중복 수행을 야기 시킬 수도 있으며, 특정한 한 가지 응답 방법을 사용하는 경우, 다른 노드로의 에이전트 이동이 지연되어 전체 작업의 수행 또한 지연되므로, 여정에 차질을 빚을 수도 있는 것이다. 이외에도, 결과물 응답이 가지는 기능은 다양할 수 있으며, 이에 관한 자세한 사항들은 다음 장에서 기술된다. 이와 같이, 결과물 응답은 분산 환경에서 여러 가지 의미를 가질 수 있는 중요한 사항이라 할 수 있으며, 다양한 경우를 고려한 유연한 응답 방식이 필요하게 된다.

본 논문에서는 결과물 응답 시점에서 고려될 수 있는 여러 상황에 대해 유연하게 대처할 수 있도록, 고정응답(fixed reply), 동적응답(dynamic reply), 그리고 다중응답(multiple reply) 등 3가지 응답 기능을 가진 이동 에이전트 개념이 제안된다. 그리고, 이를 기반으로 하는 간단한 SCM 실험 모델이 설계 구현된다. 이는 공급 체인에서 공급자들로부터 적시에 정보를 얻어내는 것이 중요하며, 또한 협동 작업의 유형이 많아서, 경우에 따라 다르게 결과물을 응답 해야 할 필요가 있어 제안되는 응답 기능을 사용하기에 적합한 좋은 분산 응용 중의 하나이기 때문이다. 제안되는 응답 기법은 응답 대상 노드들이 우선 순위 순으로 나열된 우선순위 응답 리스트(Priority Reply List)를 사용하고 있다. 설계 구현된 시스템은 최근 가장 많이 사용되고 있는 Java 기반의 이동 에이전트 개발 플랫폼인 AGLETS[6]에 기반을 두고 있다.

본 논문은 다음과 같이 구성된다. 2장에서는 이동 에이전트 기반을 둔 SCM 관련 연구들과 본 연구의 제안 배경, 그리고 제약점 등이 기술되며, 3장에서는 유연 응답 기능을 구성하고 있는 3가지 응답 모드에 대한 내용이 상세하게 기술된다. 4장에서는 3장에서 제안된 유연 응답 기능을 가진 이동 에이전트 기반의 공급 체인 관리를 위한 간단한 실험 모델이 설계되며, 5장에서는 본 연구를 위해 구현된 이동 에이전트 실험 및 실험 환경이 기술되며, 마지막으로 6장에서 결론 및 향후 연구가 기술된다.

2. 관련 연구 및 제안 배경

이동 에이전트 개념이 가상 기업을 대상으로 그들 간의 공급 체인의 효율적 관리와 정보 기술 시스템들과의 연계를 위한 연구에 활발히 적용되고 있다. Rabelo 등은 식품산업에 있어서 공급 체인 관리를 위한 이동 에이전트 기반의

모니터링 시스템에 관한 연구를 발표하였으나, 초기 연구 단계로 시스템 설계를 위한 방향만 제시하고 있으며[12], Papaioannou 등은 이동 에이전트 기술을 가상 기업간의 효율적인 정보처리에 적용하기 위한 이동 에이전트 논리를 제안하였으며, 그것을 바탕으로 간단한 이동 에이전트 모형을 설계하였다[13-14]. 또한 Brugali 등은 기업간의 공급 체인 관리에 이동 에이전트 기술을 적용하였는데[15], 이들 대부분은 공급 체인 관리에 이동 에이전트 기술을 적용하기 위한 모델의 정립, 설계 그리고 방향에 대한 설정 등에 관한 연구가 대부분이다. 그리고, 최근에 에이전트의 중복 수행 방지[19-20], 에이전트 규모를 고려한 네트워크의 효율적 사용[21], 작업 수행 결과물 응답의 유연성[22]에 관한 연구들이 이루어지고 있다. 본 논문에서 다양한 환경을 고려한 유연한 응답 기능을 제안하는 배경은 다음과 같다.

첫째, 다양한 미디어 제작 기술의 발달로, 분산 환경에서 작업 수행 후 얻어지는 결과물은 그 규모가 적을 수도 있으나, 영상, 음악, 그리고 음성처럼 그 규모가 부담스러운 경우도 자주 존재하고 있다. 후자의 경우, 결과물 전송 전에 응답 대상 노드로의 접속 가능성 을 파악한 후에 결과물을 전송하는 것이 네트워크 사용에 효율적이라 할 수 있다.

둘째, 기존의 클라이언트/서버, RPC, 이동 에이전트 등 대부분의 경우, 수행 결과물 전송이 작업을 요청한 노드로 이루어지기도 하지만, SCM과 같은 분산 응용에서는 응답이 모든 경우에 있어서 특정한 노드로만 이루어지는 것이 아니라는 것이다. 즉, 작업 특성에 따라, 예를 들면 비밀유지가 이루어져야 하는 작업이라면, 반드시 특정한 한 노드로 그 결과를 전송해야 하겠지만, 그 응답 대상 노드는 반드시 작업을 요청한 노드가 아닐 수도 있다. 또한, 어떤 그룹의 구성원들이 서로 협동하는 작업이라면, 둘 이상의 노드로 동시에 결과물을 전송 해주어야 하는 경우도 존재한다. 그리고, 구성원 중 어느 한 노드만 작업결과를 응답받으면 되는 그러한 경우도 존재할 수 있는 것이다. 예를 들면, 네트워크 상의 불법 파일을 탐색하는 응용의 경우처럼[23], 그 탐색 결과는 오히려 지역적으로 분산되어 있는 여러 감시원 노드들 혹은, 신속성을 필요로 한다면, 그들 중 가능한 어느 한 노드로 신속하게 결과물을 전송해 주고 다음 작업을 위하여 다른 노드로 이동하는 것이 효율적일 경우도 있는 것이다.

세째, 작업 결과물을 응답해 주어야 할 시점에서, 어떤 이유로 인해, 특정 응답 대상 노드와의 접속이 실패하거나 지연되어 결과의 획득에 상당한 지연과 대기시간이 발생하는 경우이다. 특히, 네트워크를 통해 분산된 공급 체인을 구성하는 여러 구성 요소들 사이에서 적시에 정보를 얻어내는 것이 비용 절감과 효율적인 서비스를 요구하는 소비자와 기업 모두에게 중요한 요소로 작용하는 SCM과 같은 분야에서는 접속 지연 혹은 실패로 발생하는 응답 지연은 해결되어야 할 필요한 사항이라 할 수 있다. 이러한 경우, 네트워크의 분할(partition) 혹은 노드 크래쉬 등을 고려하여, 서로 다른 영역의 노드들을 응답 대상 노드로 지정해 둔다면, 결과물에 대한 응답 수신의 기능성과 이동의 신속성을 높일 수 있어, 해당 에이전트가 다음 작업을 가능한 빨리 시작할 수 있도록 할 수 있는 것이다. 그러므로, 다양한 경우를 고려한 응답 기법을 제공해 준다면, 필요에 따라 유용하게 이용될 수 있을 것이다.

본 논문에서 주요하게 다루고 있지는 않지만, 결과물 응답이 지니는 또 다른 의미를 들자면 작업의 수행 여부의 파악이라고 할 수 있다. 직접, 혹은 간접적인 방법을 통하여 일단 응답이 확인된다는 것은 해당 작업이 수행되었다는 것을 의미하고 있기 때문이다. 그리하여, 추후 발생할지도 모르는 중복 수행을 방지할 수 있는 척도로서의 역할도 할 수 있을 것이다.

3. 유연 응답 기법

본 논문에서 제안되는 유연 응답 기법은 고정 응답모드(fixed reply mode)와 동적 응답모드(dynamic reply mode), 그리고 다중 응답모드(multiple reply mode) 세 가지 모드로 구성된다. 이들 3가지 모드는 응답 대상 노드의 수와 선정 방식에 의해 구분되어 지며, 선정 범위는 에이전트에게 주어지는 PRL로 한정된다.

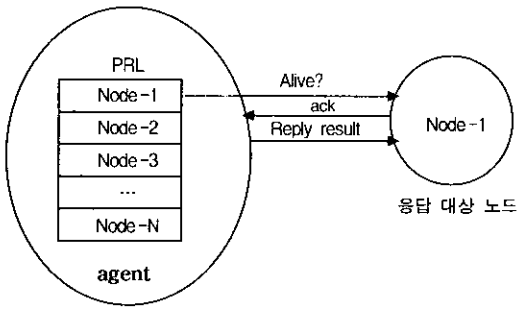
3.1 고정 응답 모드(Fixed Reply Mode)

고정 응답모드(FRM)는 응답 대상 노드가 특정한 하나의 노드로 지정되는 경우의 응답 형식이다. 그 특정 노드는 PRL 상에서 우선순위가 가장 높은 첫 번째 엔트리에 지정되며, 그 것이 고정 응답 대상 노드이다. 그 외의 엔트리들은 무시한다. (그림-1)이 보여주고 있는 것처럼 작업 노드에서 작업을 마친 에이전트는 PRL의 첫 번째 엔트리, 즉 고정 응답 대상 노드에게로만 결과물을 전송한다. 그 과정은 다음과 같다.

1) PRL의 첫 번째 노드로 접속 가능 여부 메시지(ALIVE)를 전송하고 타이머를 구동 한다.
2) 설정된 시간 내에 ACK가 수신되면 결과물을 전송하고, 다음 노드로 이동한다 그렇지 않으면, 1) 부터 반복한다.

그러므로, FRM의 경우 응답 대상 노드가 접속 가능할

때까지 기다리며 결과물이 전송되지 않는 한, 다음 노드로의 이동은 일어나지 않는다. 응답 대상 노드로의 접속이 가능하지 않으면, 가능할 때까지 결과물의 전송이 일어나지 않으며, 동시에 다음 노드로의 이동도 일어나지 않아, 작업 지연이 커질 수 있다는 단점이 있으나, 에이전트의 작업 수행 여부를 파악하는 데는 유용한 응답 기법이라 할 수 있다. (그림 1)은 FRM을 사용하는 정상적인 경우를 보여주고 있으며, 접속 불가능한 경우는 동일한 동작의 반복이다. 여기서, Node-1이 PRL의 첫 번째 엔트리 주소로, 고정 응답 대상 노드이다. 사용자는 작업 에이전트 생성 시, 특정 응답 대상 노드의 주소를 PRL 상의 첫 번째 엔트리에 지정해 주어야 한다. 이때, 이 주소는 에이전트를 생성한 소스 노드일 경우도 있으나, 그렇지 않을 수도 있다. FRM 응답 기법은 공개하기 어려운 작업 결과물을 특정 노드로만 응답을 해야만 하는 경우, 사용할 수 있는 응답 기법이다.



(그림 1) 정상적인 경우의 FRM 응답 과정

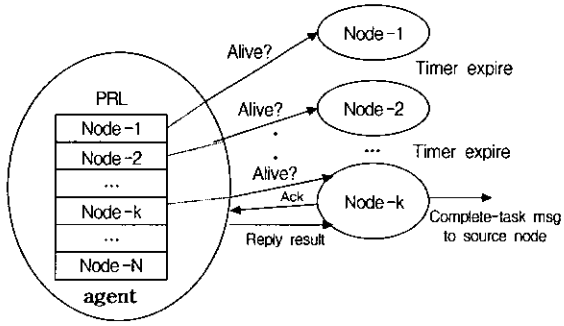
3.2 동적 응답 모드(Dynamic Reply Mode)

FRM 응답 방식이 PRL 상의 첫 번째 엔트리를 응답 대상 노드로 하는 반면, 동적 응답모드(DRM)는 응답 대상 노드는 하나이지만, 그 선정 범위가 PRL 상의 모든 노드들이라는 것이 다르다. 응답 대상 노드의 선택은 우선 순위 차례로 접속 가능성을 조사하면서 가장 먼저 접속 가능한 노드를 응답 대상으로 선정하여 결과물을 전송한다. 그러므로, FRM이 응답 대상 노드가 접속 불가능 한 경우, 접속 가능할 때까지 결과물 전송이 지연되는 반면, DRM은 설정된 시간 이내에 접속이 가능하지 않으면 다음 순위의 노드를 응답 대상 노드로 하여 접속 가능성을 조사하는 과정을 반복하면서, 가장 먼저 접속 가능한 노드를 응답 대상으로 선정하여, 결과물을 전송한다. 이때, 설정된 시간 내에 접속이 되지 않아, TimeOut이 발생한 노드는 PRL에서 삭제된다. 그것은 설정된 시간 내에 접속이 가능하지 않다는 것은 여정 동안 수행되는 작업에 대해서도 접속될 가능성이 적으며, 그리고 다음 작업에 대한 결과물 응답 시 조사시간의 절약과 결과물의 분산을 가능한 줄이기 위해서 이다. 그러나, 에이전트를 생성한 소스 노드는 원래 PRL을 유지하고 있다. 그 과정을 요약하면 다음과 같다.

1) PRL에 있는 첫 번째 엔트리 노드를 현재 노드로 지정한다.
2) 현재 노드로 접속 가능 여부 메시지(ALIVE)를 전송하고 타이머를 구동한다.
3) 설정된 시간 내에 ACK가 수신되면 현재 노드로 결과물을 전송하고, 다음 노드로 이동한다.
4) 그렇지 않으면, 현재 노드를 PRL로부터 삭제하고, PRL 상의 다음 노드를 현재 노드로 지정하고 2) 부터 반복한다.

예를 들어, 특정 물품들의 가격 조사 결과물을, 자신의 상급자들 중 어느 누구에게든 빨리 보내 주어야 하는 경우를 생각해보자. 사용자는 결과물을 수신할 수 있는 상급자들을 우선순위로 PRL에 지정해 놓으면, 에이전트는 작업을 완료한 후, 상급자 그룹 멤버 노드들을 차례로 탐색하며 접속 가능한 첫 번째 상급자에게 그 결과물을 전송할 것이다.

DRM은 주어진 PRL 상의 첫 번째 엔트리부터 순차적으로 접속을 시도하여 접속 가능한 처음 노드로 결과물을 전송하고 다음 노드로 이동한다. 그러므로 FRM에 비해 응답 지연을 줄여, 에이전트의 이동이 신속하게 이루어지게 할 수 있다는 장점을 가진다. DRM의 경우, 선정된 응답 대상 노드가 소스 노드가 아닌 경우, 소스 노드가 생성하여 보낸 에이전트로부터 결과물을 자신이 수신하였다는 사실을 통보하여 주든지, 혹은 소스 노드 자신이 수신 여부를 확인하든지 하는 과정이 필요할 수 있다. 전자의 경우에는 결과물을 수신한 노드가 소스 노드를 특정 노드로 정해 고정 모드 응답 방식으로 결과물 수신에 대한 확인 메시지를 전송하거나 하는 방법을 통해 이루어질 수 있으며, 후자의 경우, 기존의 시스템은 확인할 방법이 여정 리스트를 통해 방문 확인 밖에는 할 수 없으나, 본 응답 기법에는 PRL 자료가 존재하므로, 문제가 복구된 후, 소스 노드가 PRL 상의 노드들을 조사함으로써 결과물 수신 여부를 확인할 수 있다. 이러한 DRM 동작 과정을 보여주고 있는 것이 (그림 2)이다. 여기서 각 메시지에 대한 시간 흐름은 보여주고 있지 않으며, 그에 관한 사항은 4장의 동작 모델 설계에서 시퀀스 다이아그램으로 보여주도록 한다((그림 7) 참조).

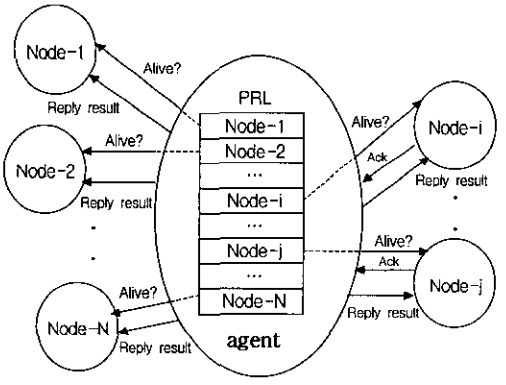


(그림 2) Node-k에서 응답이 온 경우의 DRM 응답 과정

작업 특성에 따른 응답 방식의 하나로 DRM을 사용할 수도 있으나, 네트워크 분할 혹은 노드 크래쉬 등으로 특정 노드로의 접속이 실패하는 경우, 기다릴 수 없는 상황에서 신속한 에이전트의 이동을 위한 간접 응답의 방법으로도 사용될 수 있다. 또한 결과물의 크기가 부담스러운 경우, 응답 대상을 늘려 응답 실패를 줄이는 방식으로 사용할 수 있다. 다른 측면에서 보면, 작업 수행 여부의 파악에 대한 가능성을 높일 수 있는 기법이기도 하다. 왜냐하면 PRL 상에 최소한 하나 이상의 도달 가능한 노드만 존재하면 응답을 받을 수 있으며, 이동 에이전트는 다음 노드로 이동을 하여 작업을 수행할 수 있기 때문이다. 여기서 응답 대상 노드에 대한 접속 가능성 여부를 파악하기 위한 방식으로 2가지가 있을 수 있다. 하나는 Pre-Check/Post-Sending(PCPS)이며, 다른 하나는 Pre-Sending/Post-Check(PSPC)이다. PCPS는 노드의 접속 가능성 여부를 먼저 조사한 후에 결과물을 전송하는 방식이며, PSPC는 결과물을 먼저 전송하고 결과의 수신 여부를 확인하는 방식이다. 본 논문에서는 PCPS를 사용하고 있는데, 이는 결과물 용량이 클 경우 채널의 점유율이 커져 접속 가능하지 않을 경우 채널의 낭비가 있을 수 있으므로, 먼저 경량급 메시지를 통하여 접속 가능성을 파악한 후에 결과물을 전송하는 것이 채널의 사용에 있어서 효율적일 수 있기 때문이다. Pre-Check 메시지로는 ALIVE/ACK를 사용하고 있다. PRL 상의 응답 노드에게 ALIVE 메시지를 전송하여 설정된 시간 안에 동작여부 메시지 ACK를 수신하는 경우 작업 결과를 해당 노드로 송신하게 된다. 최대 응답 대기시간은 에이전트 생성시 사용자에 의해 결정된다(기본 값은 100msec). 그리하여, 최대 응답 대기 시간 내에 ACK 메시지를 수신하지 못하면 에이전트는 TimeOut이 발생하여 다음 노드의 접속 가능성 여부를 조사한다. 이러한 방법을 통해 PRL 상의 노드들을 순차적으로 조사하여 ACK 메시지를 응답하는 첫 번째 노드로 작업 결과를 송신하고 에이전트는 다음 작업을 위해 다른 노드로 이동하거나, 더 이상 이동할 노드가 없으면, 즉 여정에 따라 모든 작업을 완료하면 소멸된다. 그러므로, PRL 상에 노드들을 지정하는 경우, 가능한 물리적으로 서로 높은 분리성을 가지도록 또는 부분적으로 네트워크 독립성을 유지하도록 PRL을 구성하면 응답 수신 가능성을 더욱 높일 수 있을 것이다.

3.3 다중 응답 모드(Multiple Reply Mode)

FRM이나 DRM은 응답 대상이 하나인 반면, 다중 응답 모드(MRM)는 응답 대상 노드가 2개 이상이며 선정 범위는 PRL 상의 모든 노드들인 경우이다. SCM 혹은 불법 파일 탐색[23]과 같은 협동작업과 같은 응용에서 그룹 단위로 작업 결과물을 수신해야 하는 경우 유용하게 사용될 수 있는 응답 기법이다. 예를 들어, 특정 물품들에 대한 가격 정보를 얻어 그 결과물을 여러 곳으로 전송해야 하는 경우라든지, 파일 탐색 결과를 여러 곳으로 전송해야 하는 경우를 생각해보면 MRM의 사용 경우를 알 수 있을 것이다. 이때, PRL은 응답 대상 노드들의 그룹이다. DRM을 PRL 상의 모든 노드들을 대상으로 확장 시킨 것과 유사하나, 응답 대상 노드의 접속 가능성 조사를 위한 방법이 DRM과는 다르다. DRM에 있어서는 PRL 상의 각 노드에 대해 순차적으로 접속 가능성을 조사하였으나, MRM의 경우는 PRL 상의 모든 노들에 대해 동시에 접속 가능성을 조사한다는 것이다. 즉, PRL 상의 모든 응답 대상 노드들에게 ALIVE 메시지를 전송한 후 타이머를 구동시킨다. 그리하여 설정된 시간 내에 k개 이상의 ACK 메시지를 수신하면 에이전트는 그 후의 ACK와 무관하게 모든 응답 대상 노드들에게 결과물을 응답한다. 여기서 $k \geq 2$이며, 에이전트 생성자에 의해 결정된다. 그러나, 설정된 시간 내에 k개 이상의 ACK 메시지를 수신하지 못하게 되면 타이머는 재 구동되고 전과 같은 동작이 반복되게 된다. 그 과정(k = 2)을 보여주고 있는 것이 (그림 3)이다. 2개 이상의 다중 노드로 결과물을 전송하기 위한 응답 방식이다.



(그림 3) Node-i와 Node-j에서 ACK가 온 경우의 MRM 응답 과정

본 연구에서는 상기 3가지 응답 모드를 지원하기 위하여, ReplyByMode 클래스를 제공하고 있으며, 또한 fixedMode, multipleMode, 그리고 dynamicMode 등의 응답과 관련된 다양한 API 메소드를 제공하고 있다. 상기 3가지 응답 모드를 결합시킨 동작 과정에 대한 알고리즘적 기술을 하면 다음과 같다.

ReplyByMode(mode, k, result)
(STEP-1) 응답모드를 확인한다
(STEP-2) 확인된 응답모드에 따라 :
2.1 고정 응답모드인 경우 :
1) PRL의 첫 번째 노드로 ALIVE 메시지를 전송하고 타이머를 구동 한다
2) 만일, 설정된 시간 내에 ACK가 수신되면 결과물을 전송하고 (STEP-3)로 간다
3) 그렇지 않으면, STEP 2.1의 1)로 간다

2.2 동적 응답모드인 경우 :
1) PRL에 있는 처음 노드를 현재 노드로 지정한다
2) 현재 노드로 ALIVE 메시지를 전송하고 타이머를 구동한다
3) 만일, 설정된 시간 내에 ACK가 수신되면 현재 노드로 결과물을 전송하고, (STEP-3)으로 간다
4) 그렇지 않으면, 현재 노드를 PRL로부터 삭제하고 다음 노드를 현재 노드로 지정하고 STEP 2.2의 2)로 간다
2.3 다중 응답모드인 경우 :
1) PRL 상의 N개의 모든 노드로 ALIVE 메시지를 보내고 타이머를 구동한다.
2) 만일, 설정된 시간 내에 k개 이상의 ACK가 수신되면 PRL 상의 모든 노드들에게 결과물을 전송하고 (STEP-3)으로 간다
3) 그렇지 않으면, STEP 2.3의 1)로 간다
(STEP-3) 여정 리스트 상의 다음 작업 노드로 이동한다.

3.4 유연 응답 기능을 위한 전제 사항

응답 모드는 에이전트 생성 시, 사용자에 의해 결정된다. 그리고, 생성된 에이전트는 자신이 생성된 소스 노드의 주소, 자신의 id, 응답 대상 리스트인 PRL 그리고 여정을 위한 여정 리스트 등의 정보를 지니고 움직인다. 또한 결과물 전송 전에 에이전트는 응답 대상 노드로의 접속 가능성을 조사한 후에 결과물을 전송하며, 지정된 시간 내에 파악이 되지 않는 경우의 동작은 각 응답 모드에 따라 다르다. 이를 위해, 유연 응답 기법은 타이머를 사용하고 있다.

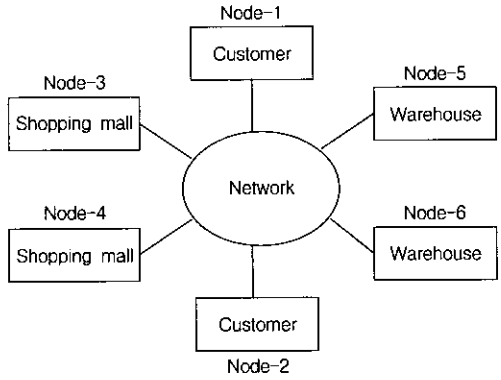
유연 응답 기법을 위한 몇 가지 전제 사항은 다음과 같다.

1) 주어진 여정을 통과해서 이동하는 에이전트는 모두 동일한 작업을 한다.
2) 여정 리스트는 최소 2개 이상의 노드를 포함하며, 최소한 하나는 이동 가능하며, 에이전트는 동적 라우팅을 통해 이동 한다. 동적 라우팅은 AGLETS이 지원하는 기능이다.
3) 사용자에 의해 주어지는 PRL 상의 노드 중 최소한 하나는 접속 가능 하다는 것을 전제로 한다. 그것은 문제 해결을 단순화 시킬 수 있으며, 그러한 사항을 전제로 하더라도 실제 상황에서 크게 어긋나지 않는다고 생각된다.
4) 접속 가능 조사에 있어서, ALIVE 메시지에 대한 ACK를 받으면, 결과물 전송이 완료될 때까지 접속은 유지된다.
5) 결과물 취합은 일어나지 않으며, 다만, 결과물을 수신했다는 사실을 알도록 한다.

4. 공급 체인 관리 실험 모델 설계

4.1 시스템 모델 설계

유연 응답 기능을 이용하는 이동 에이전트 기반의 간단한 공급 체인 관리의 실험적 모델이 설계된다. 공급 체인에는 계획, 판매, 구매, 생산, 보관, 스케줄링, 광고 등 여러 가지 형태의 구성요소가 있지만, 구현의 편의를 위해 (그림 4)와 같이 3가지 유형의 업무를 담당하는 6개의 노드 (Node-1 부터 Node-6)를 설계하였다. 즉, 2개의 고객 노드 (Node-1, Node-2), 2개의 쇼핑몰 노드(Node-3, Node-4), 그리고 2개의 물류창고 노드(Node-5, Node-6)가 그것들이다. 각 노드에는 담당업무 에이전트가 상주하고 있으며, 필요에 따라 자식 에이전트를 생성할 수 있으며, 생성된 자식 에이전트는 스스로 다른 노드로 이동할 수 있으며, 다른 에이전트와 메시지를 통해 통신 할 수 있다.



(그림 4) 공급 체인 관리를 위한 실험 모델

고객 노드(Node-1, Node-2) : CustomAgent(CustA)

고객 노드를 관리하고 사용자 인터페이스를 제공하는 에이전트인 CustomAgent(CustA)가 존재하고 있으며, 쇼핑몰에서 물품을 구매하고자 하는 고객에게 인터페이스를 제공한다. CustA는 이동성이 없는 에이전트로, 고객은 이를 통해서 원하는 물품에 대한 가격 정보를 요청(inquiry)하거나 주문(order)을 할 수 있다. CustA는 사용자가 선택한 작업에 따라 이동 에이전트인 InquiryAgent(InqA) 혹은 OrderAgent (OrdA)를 생성 할 수 있으며, 사용자에 의해 선택된 응답모드, PRL, 그리고 여정을 자신이 속한 노드의 주소와 더불어 자식 에이전트에게 넘겨준다.

쇼핑몰 노드(Node-3, Node-4) : ShoppingMallAgent(ShmlA)

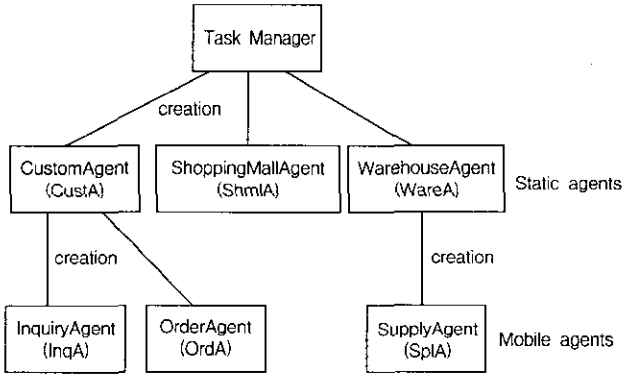
쇼핑몰 노드 관리 및 에이전트와의 메시지 통신을 담당하는 ShoppingMallAgent(ShmlA)가 존재하고 있으며 CustA와 마차가지로 이동성이 없는 정적 에이전트이다. CustA가 보낸 InqA 혹은 OrdA의 요청에 따라 물품에 대한 가격 정보를 제공하거나 혹은 물품에 대한 주문 정보를 받아 주문 관련 데이터를 갱신하는 역할을 하고, 또한 물류창고 노드의 WarehouseAgent가 보낸 작업 에이전트인 SupplyAgent가 요청한 물품에 대한 재고 및 주문량에 관한 정보를 제공한다.

물류창고 노드(Node-5, Node-6) : WarehouseAgent(WareA)

물류창고 노드를 관리하는 WarehouseAgent(WareA)가

존재하고 있으며 이동성이 없다. 사용자의 요청에 따라 이동성 작업 에이전트인 SupplyAgent(SplA)를 생성할 수 있으며, CustA의 경우와 마찬가지로, 사용자에 의해 선택된 응답모드, PRL, 그리고 여정을 자신이 속한 노드의 주소와 더불어 자식 에이전트인 SplA에게 넘겨준다. WareA는 CustA와 유사한 인터페이스를 가지며, 이 인터페이스를 통해서 작업요청을 받는다. SplA는 쇼핑몰에 존재하는 특정 물품에 대한 재고 혹은 주문량을 알아내어 적시에 쇼핑몰에 물품을 공급하기 위함이다.

위에서 정의된 각 에이전트에 대한 계층 관계를 보여주고 있는 것이 (그림 5)이다. 가장 상위에 작업 관리자(Task manager)가 존재하고 있으며, CustA, ShmlA, 그리고 WareA는 작업 관리자에 의해 생성되는 하위 에이전트들이며, InqA, OrdA는 CustA의 하위 에이전트들로 이동성을 가지고 있으며, SplA는 WareA에 의해 생성되는 하위 에이전트로 이동성을 가진다.



(그림 5) 설계된 각 에이전트들 계층도

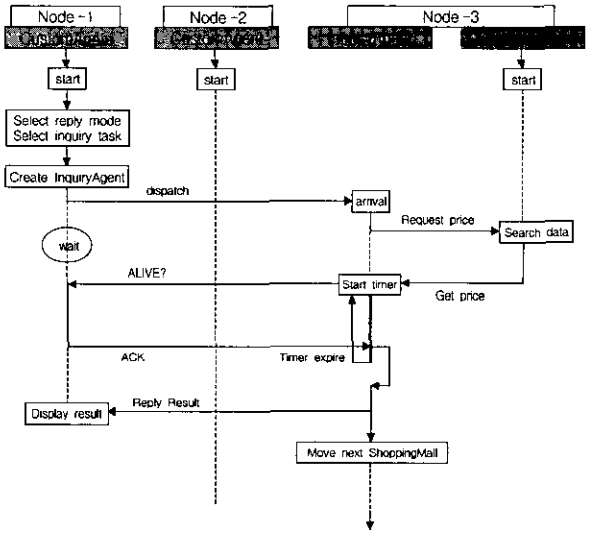
가장 상위의 작업 관리자(Task manager)는 AGLETS을 구성하는 Tahiti 서버가 그 기능을 담당하고 있는데, Tahiti 서버는 에이전트의 생성, 코드전송, 도착, retract와 같은 에이전트의 상태를 모니터링하기 위한 시스템이다. 그러므로, 각 노드의 관리 에이전트인 CustA, ShmlA, 그리고 WareA는 Tahiti 서버에서 사용자에 의해 생성되어 동작한다.

4.2 유연 응답 동작 과정 설계

설계된 각 에이전트들 사이에 존재하는 결과물 응답 경우는 CustA와 InqA 사이에 FRM, DRM, MRM 등 3경우, CustA와 OrdA 사이에 3경우, 그리고 WareA와 SplA 사이에 3경우, 총 9가지 경우가 있을 수 있으나, 각 에이전트 사이의 3가지 응답 과정은 서로 유사하므로, 여기서는 CustA와 InqA 사이의 일어나는 각 3가지 경우의 응답 동작에 관해서만 기술하기로 한다. 단, PRL에는 Node-1과 Node-2의 주소가 지정되어 있으며, 여정 리스트에는 Node-3와 Node-4가 지정되어 있다. 그리고, InqA는 Node-1의 CustA에 의해 생성된다.

4.2.1 고정 응답모드(FRM)의 동작 설계

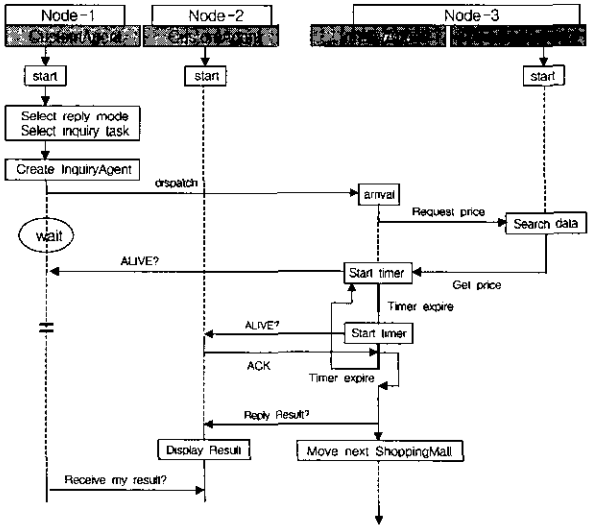
CustA(Node-1)와 InqA 사이에 FRM 응답 방식을 사용하여 결과물 응답을 하는 동작 과정를 요약적으로 보여주고 있는 것이 (그림 6)이며, 생성 노드인 Node-1의 CustA가 접속 가능한 경우를 보여주고 있다.



(그림 6) CustA와 InqA 사이에 FRM 기반의 응답 과정

4.2.2 동적 응답모드(DRM)의 동작 설계

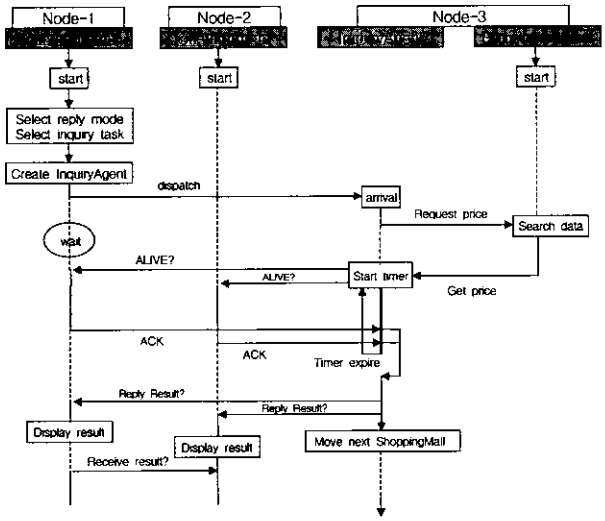
CustA(Node-1)와 InqA 사이에 DRM 응답 방식을 사용하여 결과물 응답을 하는 동작 과정을 요약적으로 보여주고 있는 것이 (그림 7)이다. (그림 7)은 Node-1이 응답 시점에 접속 불가능한 경우를 가정하고 있으며, 이러한 접속 불가능 환경은 Node-1의 CustA가 Alive 메시지에 대해 응답을 하지 않는 행위로 모델화 할 수 있다. 추후, Node-1의 CustA는 Node-2가 결과물을 수신했는지를 PRL을 기반으로 Node-2의 CustA에게 확인 요청할 수 있다.



(그림 7) CustA와 InqA 사이에 DRM 기반의 응답 과정

4.2.3 다중 응답모드(MRM)의 동작 설계

CustA와 InqA 사이에 MRM 응답 방식을 사용하여 결과물 응답을 하는 경우를 요약적으로 보여주고 있는 것이 (그림 8)이다. (그림 8)에서는 Node-1과 Node-2의 각 CustA가 접속 가능하다는 응답을 보내준 경우를 보여주고 있다. 마찬가지로, Node-1의 CustA는 Node-2가 결과물을 수신했는지를 PRL을 기반으로 Node-2의 CustA에게 확인 요청할 수 있다.



(그림 8) CustA와 InqA 사이에 MRM 기반의 응답 과정

4.3 유연 응답 동작을 위한 클래스 객체 설계

본 논문에서 제안된 유연 응답 동작의 중심이 되는 ReplyByMode 클래스와 SCM 응용을 위한 실험 에이전트들과 관련된 여러 가지 메소드들을 포함하는 SimpleSCM 클래스가 설계 구현된다. ReplyByMode 클래스가 제공하는 각 메소드에 대한, 명칭, 리턴 값의 유형, 그리고 사용하는 매개변수들에 관한 요약이 <표 1>에 기술되어 있다. 이 클래스의 중심 데이터는 PRL이므로 이와 관련된 메소드들이 대부분이다. 크게 구분하면, 응답 모드 설정, PRL에 기반을 둔 응답 방식들, 그리고 PRL 구성 기능들의 3종류로 구분할 수 있다. SimpleSCM 클래스가 제공하는 메소드들은 <표 2>에 같은 방법으로 요약 기술되어 있으나, 이 클래스에 대해서는 커다란 의미를 부여하기 어렵다. 단지, 본 실험 모델에서 사용되는 에이전트들이 공통으로 사용하는 간단한 메소드들의 모임이며, 얼마든지 다른 형태로 구성될 수 있다. 이렇게 함으로써 본 실험에서 설계된 각 에이전트는 몇 가지 자신의 고유 작업과, <표 1>과 <표 2>에 기술되어 있는 메소드들을 상속 받거나 하여 매우 용이하게 구현될 수 있다. 그러므로, 실험에서 구현된 각 에이전트들은 데이터의 초기화와 작업 및 제어 등에 관한 사항을 제외하면 새로이 사용하는 메소드들은 별로 없는 매우 간단한 구조를 가질 수 있도록 설계 하였다.

<표 1> ReplyByMode 클래스를 구성하는 메소드

No	Method Name	Type	Arguments
1	setReplyMode	void	(String replyMode)
2	sendReply	void	(Message result, long duration)
3	fixedMode	void	(Message result, long duration)
4	multipleMode	void	(Message result, long duration)
5	dynamicMode	void	(Message result, long duration)
6	sendAlive	void	(Message result, long duration)
7	getSizePRL	int	none
8	getUrlHostNode	String	(int priority)
9	getPriorityPRL	int	(String rrl)
10	getPRL	Enumeration	(String prlName, String url)
11	addPRL	void	(String prlName, String url)
12	insPRL	void	(Strig prlName, String url, int priority)
13	delPRL	void	(String prlName, int priority)

<표 2> SimpleSCM 클래스를 구성하는 메소드

No	Mothod Name	Type	Arguments
1	selectProduct	void	none
2	selectTaskInfo	void	none
3	selectReplyMode	void	none
4	selectTimeOut	void	none
5	selectChart	void	none
6	displayResult	void	(Message result)
7	displayChart	void	(Message result)
8	sendAck	void	(Message Ack)
9	sendRequest	void	(String productName, String requestInfo)
10	searchProductInfo	void	(String productName, String requestInfo)
11	sendSearchInfo	void	(Message result)
12	updateInfo	void	(String productName, int data)

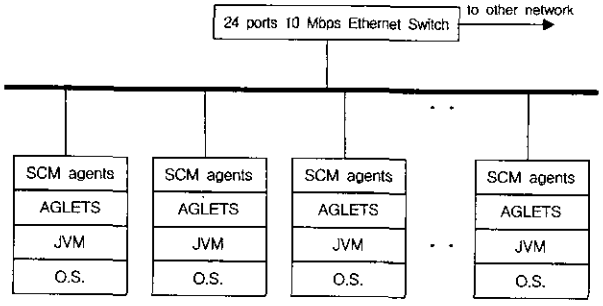
5. 실험 환경 및 모델 구현

5.1 실험 환경

AGLETS은 그가 가지는 많은 장점 때문에 최근 여러 응용에 가장 많이 이용되고 있는 Java 기반의 이동 에이전트 플랫폼으로, 본 시스템이 기반으로 하고 있는 플랫폼이다. AGLETS은 에이전트 관리를 위한 Tahiti 서버와 각종 API로 구성되어 있다. AGLETS의 API 패키지는 이동성을 위한 자바 메소드들의 모임이고, Tahiti 서버는 에이전트의 생성, 코드전송, 도착, retract 같은 에이전트의 상태를 감시하기 위한 시스템이다. AGLETS의 기존 API에 유연 응답 기능을 위한 메소드들이 추가되었다. AGLETS의 장단점은 기본적으로 Java의 장단점과 같다.

본 연구를 위한 실험 환경은 6대의 PC 노드들로 구성되어 있으며, 각 노드는 10Mbps 이더넷 스위치를 통해 외부와 연결되어 있다. 각 노드는 JVM과 확장된 API를 가지는 AGLETS 플랫폼과 공급 체인 관리 응용 에이전트(SCM agents)를 가지고 있다. 그 구성도를 보여주고 있는 것이 (그림 9)이다.

(그림 9)에서 기술된 SCM agents는 본 연구에서 설계된 에이전트들로 CustA, ShmIA, WareA, InqA, OrdA, 그리고 SplA들의 집합을 의미한다. 각 노드의 기능 상, Node-1과 Node-2에는 CustA, InqA, 그리고 OrdA가, Node-3와 Node-4



(그림 9) 실험 시스템의 구성도

에는 ShmlA, 그리고 Node-5와 Node-6에는 WareA와 SplA가 설치되어야 하나, 편의를 위해 모두 설치하고 작업 관리자인 Tahiti 서버에서 해당 작업을 생성 구동 시키는 것으로 하였다.

5.2 에이전트 구현

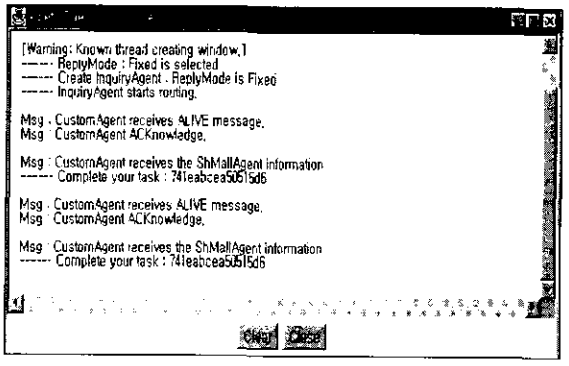
앞에서도 언급하였듯이, 설계된 ReplyByMode 클래스와 SimpleSCM 클래스를 기반으로 하면, 본 실험에서 설계된 각 에이전트는 데이터의 초기화와 작업 및 제어 등에 관한 사항을 제외하면 새로이 사용하는 메소드들은 별로 없는 매우 간단한 구조를 가지게 되어, 매우 용이하게 구현될 수 있다. 또한 WareA의 경우, CustA와 하는 일이 작업의 종류와 결과의 표현을 그래픽으로 하는 것만 다르므로 인터페이스만 약간 다르며 다른 사항은 동일하다. 그리고 이동 에이전트들인 InqA, OrdA, 그리고 SplA들도 ReplyByMode 클래스의 메소드들과 기타 몇 가지 메소드로 구성되며, 요청하는 작업만 다를 뿐 나머지는 동일하다. 예를 들면, InqA는 sendRequest() 메소드를 사용하는데, 물품 명칭(본 실험에서는 A, B, …, I)과 요청 정보인 requestInfo("pri", "ord", ora", "stk", "sal") 중 "pri"(가격요청)를 사용하며, OrdA는 "ord"(주문요청)를 사용한다. 그리고 WareA에 의해 생성된 SplA는 "ora"(주문량 요청), "stk"(재고량 요청), 그리고 "sal"(판매량 요청)을 사용하는 것이 다를 뿐 나머지 동작들은 동일하게 구성된다.

5.3 응답 모드 실험

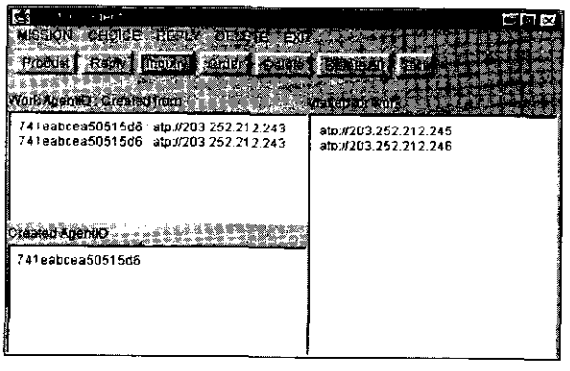
유연 응답 기능에 의한 결과물 응답의 경우는 여러 경우가 존재하고 있으나, 대표적으로 고객 노드인 Node-1과 Node-2, 그리고 쇼핑몰 노드인 Node-3와 Node-4 간의 작업에 있어서 각 응답 과정을 보여주기로 한다. 그 과정은 (그림 6), (그림 7), 그리고 (그림 8)에서 설계된 동작과정을 따르고 있으며, 실제로는 필요하지 않은 메시지이지만, 각 단계를 보여주기 위하여 사용자에 의한 사건에 관한 메시지와 진행 과정을 보여주기 위한 메시지들을 간격을 띄어 구분하여 보여주고 있다. 이러한 메시지 출력은 각 노드의 Tahiti 서버 창을 이용하고 있다.

5.3.1 고정 응답모드(FRM)의 경우의 동작 실험

Node-1의 CustA와, CustA에서 생성되어 쇼핑몰 노드인 Node-3, Node-4를 여정으로 가지고 이동된 InqA 간의 주요 동작 및 결과물 응답 과정을 보여주고 있는 것이 (그림 10)이다. (그림 10)의 (a)는 Node-1의 Tahiti 서버 창으로, CustA에서 사용자에 의해 응답 모드로는 FRM이 선택되었으며, 수행 작업으로는 InqA가 선택되어 이동되는 과정, ALIVE 메시지 수신, 그리고 그에 대한 acknowledge과정 등이 (그림 10)의 (a)에 나타나 있으며, (b)는 CustA의 사용자 인터페이스 창이며, 자신에 의해 생성된 InqA의 id와 결과물을 응답한 작업 에이전트인 InqA의 id, 그리고 작업이 수행된 노드, 즉 Node-3와 Node-4의 주소가 나타나 있다. (그림 10)의 실험은 InqA가 정상적으로 2개의 쇼핑몰 노드로 구성된 여정을 완료하였음을 보여주고 있으며, PRL 처음 엔트리에 자신을 생성한 CustA(즉, Node-1)의 주소가 지정되어 있는 경우이다.



(a) Node-1의 Tahiti 서버 창

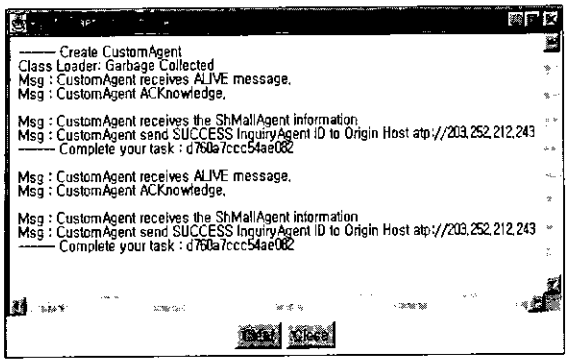


(b) Node-1의 CustA의 창

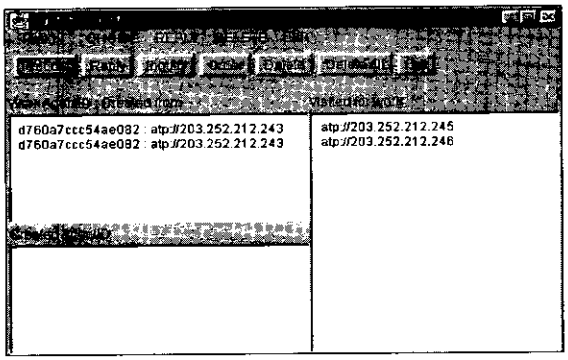
(그림 10) CustA와 InqA 간의 FRM을 사용하여 동작하는 과정

5.3.2 동적 응답 모드의 경우의 동작 실험

(그림 11)은 응답 모드만 DRM으로 수정한 경우의 동작 과정을 보여주고 있는 그림이다.이다. 단, DRM에 있어, Node-1의 CustA가 설정된 시간 내에 응답을 하지 않아 TimeOut이 발생하여, 결과물을 Node-2의 CustA로 응답하는 경우이다.



(a) Node-2의 Tahiti 서버 창

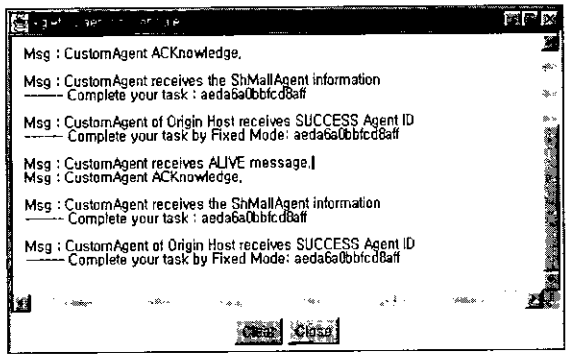


(b) Node-2의 CustA의 창

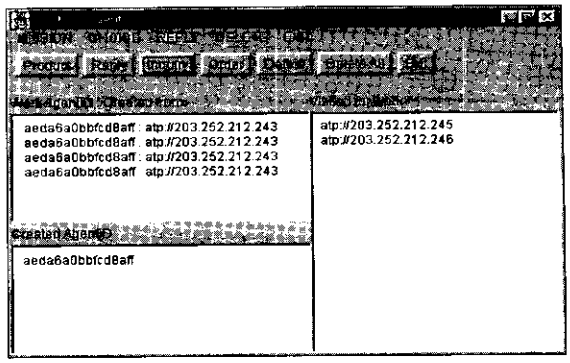
(그림 11) CustA와 InqA 간의 DRM을 사용하여 동작하는 과정

5.3.3 다중 응답모드(MRM) 경우의 동작 실험

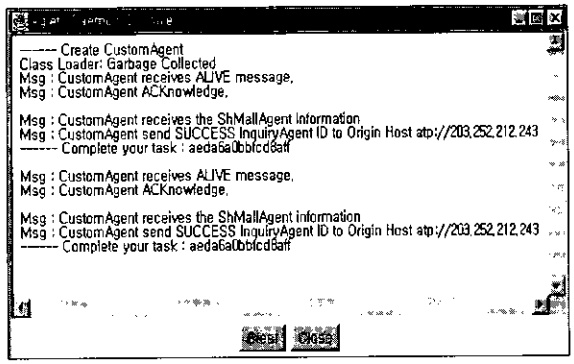
(그림 12)는 MRM의 동작 과정을 보여주고 있는 그림이다. (그림 12)의 (a)와 (b)는 Node-1의 CustA의 수행과정을 보여주고 있으며, (c)와 (d)는 Node-2의 CustA의 수행과정을 보여주고 있다. (그림 12)의 실험은 PRL 엔트리에는 Node-1과 Node-2의 주소가 지정되어 있고, 결과물을 응답받는 최소 노드 수 k=2이고, Node-1의 CustA와 Node-2의 CustA가 정상적으로 수행되는 경우이다. Node-3, Node-4의 ShmlA로부터 정보를 수신한 InqA는 CustA와 CustA로 ALIVE 메시지를 전송하고, 수신 메시지ACK를 두 CustA로부터 받아 응답 가능한 노드의 수가 2이므로, 결과물을 Node-1의 CustA와 Node-2의 CustA로 응답한다.



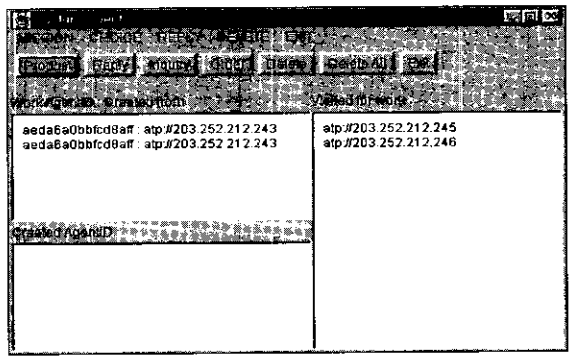
(a) Node-1의 Tahiti 서버 창



(b) Node-1의 CustA의 창



(a) Node-2의 Tahiti 서버 창



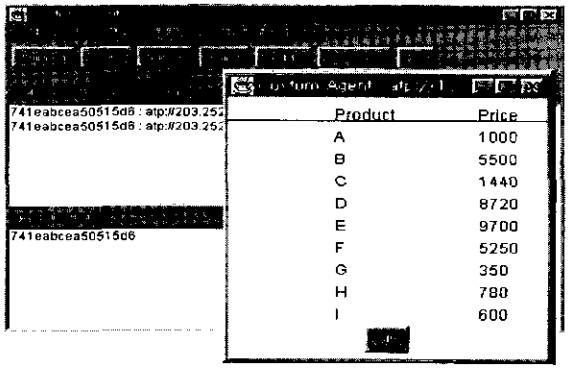
(b) Node-2의 CustA의 창

(그림 12) CustA와 InqA 간의 MRM(k=2)을 사용하여 동작하는 과정

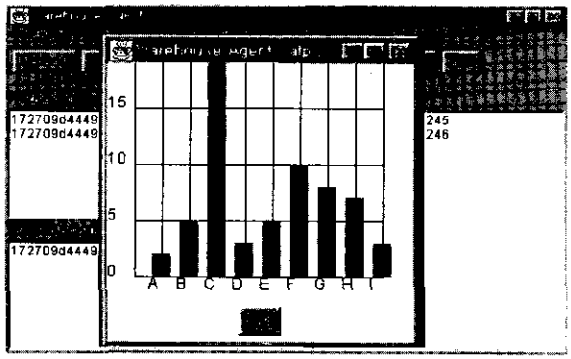
5.3.4 결과의 시각화

수신된 결과물 데이터의 이해도를 높이기 위해 본 논문에서는 필요에 따라, 텍스트, 막대, 선, 파이 그래프를 사용해서 결과를 시각화 할 수 있도록 하고 있다. (그림 13)의 (a)는 InqA가 보낸 ShmlA에 있는 물품들의 가격 정보를 CustA가 보여주는 것으로 텍스트로 표현하고 있으며, (b)는 SplA로부터 WareA로 전송된 ShmlA에서의 물품별 주문량을 WareA가 막대그래프를 사용해서 보여준 것이다. CustA 혹은 WareA가 제공하는 인터페이스를 통해서 텍스트, 막대, 파이, 그리고 선 그래프 중 하나를 선택할 수 있다. 수신된 결과물은, 편의적으로 CustA(혹은 WareA)에서

작업이 수행된 노드의 주소를 클릭함으로써 보여질 수 있도록 하였다.



(a) 가격정보 출력의 예



(b) 주문량 정보 출력의 예

(그림 13) InqA와 SplA가 전송한 결과를 CustA와 WareA에서 출력한 형태

6. 결론 및 향후 연구

이동 에이전트는 여러 가지 분산 응용에서 네트워크 부하와 지연을 극복할 수 있는 기법 중 하나로 최근 활발히 연구되고 있는 새로운 패러다임이다. 본 논문에서는 에이전트의 작업 수행 후, 작업 특성과 상황에 따라, 결과물의 유연한 응답을 지원하기 위해 우선순위 응답 리스트(PRL) 기반의 3가지 응답 방식을 제안, 구현하였으며, 이를 위한 유용한 API 메소드들을 포함하는 클래스들이 설계 구현되었다. 그리고, 이를 기반으로 공급 체인 관리를 위한 간단한 실험 시스템을 구현하였다. 제안된 응답 기능을 가진 이동 에이전트 시스템은 SCM과 같이 협동작업의 경우처럼, 다양한 결과물 분배를 필요로 하는 분산 응용에 적용되면, 유용할 수 있을 것이다.

최악의 경우에 대한 연구와, 제안된 응답 기능 및 변형 확장 시킨 응답 기능을 가지는 이동 에이전트 플랫폼을 설계, 개발 중에 있으며, 더불어 좀 더 현실적인 SCM을 위해 다양한 유형의 공급 체인 구성 요소들을 포함하는 시스템을 설계 중에 있다. 예를 들면, WarehouseAgent는 ManufacturingAgent 같은 다른 에이전트와 상호작용 할 수 있다. 예를 들어, 창고에 있는 재고량이 주문량보다 충분하지 않으면 WarehouseAgent는 공장에 있는 ManufacturingAgent에게 공급을 요청 할 수 있다. 이런 부분은 추후 확장될 것이다. 이처럼 이동 에이전트 기반의 공급 체인 모델은 그 시나리오에 따라, 제조, 고객 데이터베이스 관리, 계획 등 여러 가지 타입의 객체들을 생성함으로써 쉽게 확장될 수 있다.

Acknowledgement

본 논문의 개선점에 관해 여러 부분을 지적하여, 논문의 개선에 도움을 주신 익명의 심사위원들께 감사드립니다. 또한 프로그램 개선 작업에 많은 시간을 할애해준 ㈜아이캔텍의 강 미연 군에게도 감사를 드립니다.

참 고 문 헌

[1] Ye Chen et al., "A Negotiation-based Multi-agent System for Supply Chain Management," A working paper at dept. of CSEE, Univ. of Maryland, Jan. 1999.

[2] Ye Chen, "Using Economic Principle in MAS Study," A working paper at dept. of CSEE, Univ. of Maryland, 1999.

[3] N. Ivezic et al., "Agent-based Technologies for Virtual Enterprises and Supply Chain Management," A draft version for submission, IEEE Internet Computing, CTRC, Oak Ridge National Lab., 1999.

[4] H. J. Burckert et al., "MAS Technologies for Supply Chain Management in Virtual Enterprises," German AI Research Center, 1999.

[5] Ye Chen et al., "Mixed Negotiation Process : Adjusting Agent Autonomy in Supply Chain Management System(SCMS) Negotiation Process," A working paper, dept. of CSEE, Univ. of Maryland, 1999.

[6] D. B. Lange and M. Oshima, "Programming and Deploying JAVA Mobile Agents with Aglets," Addison-Wesley, 1998.

[7] D. D. Roure et al., "Agents for Distributed Multimedia Information Management," Proc. of 1st Int'l Conf. on the Practical Application of Intelligent Agents and Multi-agents Technology, April, 1996.

[8] L. M. Silva et al., "Using Mobile Agents for Parallel Processing," Project Report, Dept. of Engineering Information, Univ. of Coimbra, Portugal, 1998.

[9] C. Panayiotou et al., "Parallel Computing Using Java Mobile Agents," A working paper at Dept. of CS, Univ. of Cyprus, 1999.

[10] N. Minar et al., "Cooperating Mobile Agents for Dynamic Network Routing," MIT Media Lab., 1999.

[11] C. S. Hood and C. Ji, "Intelligent Agents for Proactive Fault

Detection," IEEE Internet Computing, March/April, 1998.
[12] R. J. Rabelo and L. M. Spinosa, "Mobile-agent-based Supervision in Supply Chain Management on the Food Industry," Proc. of Workshop on Supply Chain Management in Agribusiness, AGROSOFT97, Brasil, 1997.
[13] T. Papaioannou & J. Edwards, "Using Mobile Agents To Improve the Alignment Between Manufacturing and IT Support Systems," Journal of Robotics and Autonomous Systems, Vol.27, 1999.
[14] T. Papaioannou & J. Edwards, "Manufacturing Systems Integration and Agility : Can Mobile Agents Help?," Journal of Integrated Computer-Aided Engineering, 2000.
[15] D. Brugali et al., "Inter-Company Supply Chains Integration via Mobile Agents," in The Globalization of Manufacturing in the Digital Communications Era of the 21st Century : Innovation, Agility, and the Virtual Enterprise, Kluwer Academic Pub., 1998.
[16] A. Ohsuga et al., "PLANGENT : An Approach to Making Mobile Agents Intelligent," IEEE Internet Computing, July/Aug. 1997.
[17] Ravi Kalakota and Andrew B.Whinston, "Electronic Commerce," Addison-Wesley, 1997.
[18] Won-Ho. Chung et al., A Scheme of Supply Chain Management Using Mobile Agents, Proc. of 10th Joint Conf. On Communications and Information(JCCI), 2000.
[19] J. Baumann et al., Mole Concepts of a Mobile Agent System, Tech. Report, Institute for Parallel and Distributed High-Performance Computers(IPVR), Stuttgart Univ., 1997.
[20] K. Rothermel and M. Straser, A Protocol for Preserving the Exact-Once Property of Mobile Agents, Tech. Report, Institute for Parallel and Distributed High-Performance Computers(IPVR), Stuttgart Univ., 1997.
[21] D. Bhandari and A. Joshi, An XML Based Framework to Manage Agent Migration, Univ. Maryland, 1999.
[22] Won-Ho Chung et al., A Mobile Agent Scheme with Flexible Reply and Routing for Supply Chain Management, Prof. of Asia-Pacific Conf. on Communications (APCC), 2000.
[23] 정원호 외, 동적 응답 기능을 가진 이동 에이전트를 이용한 네트워크 상의 불법 파일 탐색, HCI 학술대회발표논문집, 2001.

정 원 호

e-mail : whchung@center.duksung.ac.kr
1979년 서울대학교 전자공학과(학사)
1981년 한국과학기술원 전기및전자공학과 (석사)
1989년 한국과학기술원 전기및전자공학과 (박사)
1979년~1982년 대한전선㈜
1983년~1984년 대우통신㈜
1993년 IBM T. J. Watson 연구소 방문 연구원
1989년~현재 덕성여자대학교 컴퓨터과학부 교수
관심분야 : 분산 및 병렬운영체제, 이동에이전트, 임베디드시스템

남 희 정

e-mail : hjnam@namhae.duksung.ac.kr
1994년 덕성여자대학교 전산학과(학사)
1994년~1999년 에스원㈜
2000년~현재 덕성여자대학교 전산및정보통신학과 석사과정
관심분야 : 분산 및 병렬처리, 이동 에이전트