PC 그리드 컴퓨팅에서의 서비스 디스커버리를 위한 에이전트 플랫폼에 관한 연구*

곽준원, 백주련, 고혁진, 신동렬, 김응모
성균관대학교 컴퓨터공학과
e-mail : ssmile777@ece.skku.ac.kr

Development on Agent Platform for Service Discovery in PC Grid Computing

Jun-Won Kwak, Ju-Ryon Paik, Huk-Jin Ko, Dong-Ryul Shin, Ung-Mo Kim
Dept. of Computer Engineering, SungKyunKwan University

요 약

PC 그리드 컴퓨팅(Grid Computing) 환경이란 기존의 클라이언트/서버 구조를 최대한 이용하여 텍스트 정보뿐만 아니라 컴퓨팅 파워, 데이터 저장 장치, 첨단 실험 장비 등 가용한 모든 자원들, 나아가 인력 자원들까지도 인터넷을 통해 공유하는 기술이다. 각 클라이언트는 공동으로 작업하는 프로젝트의 처리 결과를 중심이 되는 서버에 전송한다. 중심 서버에서는 각 클라이언트들의 개별적인 처리 결과를 종합하여 단일 결과를 도출한다. PC 그리드 컴퓨팅 환경하에서 분산된 모든 서비스나 장치들이 상호작용 하기 위해서는 해당 서비스가 제공하는 것이 무엇인지, 또한 자신이 원하는 자원이 어디에 위치해 있는지를 정확히 알아야만 한다. 이런 일련의 작업들을 서비스 디스커버리(Service Discovery)라 한다. 이기종 에이전트 플랫폼 간에 자원의 공유를 위해서 FIPA 에서는 서비스 디스커버리에 대한 명세서를 제안하고 있다. 본 논문에서는 PC 그리드 컴퓨팅 환경에서 이기종 에이전트 플랫폼간에 서비스 디스커버리가 이루어 질 수 있도록 연구 및 개발을 목적으로 한다.

1. 서론

최근 대부분 기업은 주요 애플리케이션 시스템만을 위한 전용 스토리지와 서버를 운영하고 있으며, 이는 시스템의 하드웨어와 소프트웨어를 분리해 개별 시스템을 운영하는 것이 더욱 간편한 방법이기 때문이다. 그러나 실제 기업 환경에서 이와 같은 방법을 사용해 시스템을 구성한 결과 스토리지의 낭비와 작업률의 저하라는 장애 요인이 증가하였고, 결국 많은 비용을 요구하는 시스템들이 기업 인프라의 상당부분을 차지하게 됐다. 또한, 개별 시스템들이 한계 용량에 크기를 맞추게 되면서 한 시스템에서 다른 시스템으로 자원을 이동시키는 일이 결코 쉽지 않은 문제가 되었다. 결과적으로 스토리지 활용률은 약 50%, CPU 의 활용률은 15~20%에 그치고 있다. 즉, 시스템만으로는 월등하게 높은 능력이 활용률 면에서는 그에 미치지 못하고 잇는 것이다.

위에 기술한 문제들의 해결을 위해 발전된 기술이 그리드 컴퓨팅(Grid Computing)이다. 그리드 컴퓨팅이란 지리적으로 분산되어 있는 고성능 컴퓨팅 자원을 네트워크로 연결하여 특정 조직과 위치에 관계없이 사용하는 컴퓨팅 방법이다[1]. 그리드 컴퓨팅은 우선 저비용의 컴포넌트를 활용하고 자원 활용 범위를 증가시킴으로써 하드웨어 비용을 낮출 수 있다. 또한, 동일한 방식으로 각각의 시스템을 구축함으로써 근로비용을 대폭 낮출 수 있으며, 중앙 관리 툴을 통해 유지 관리 및 모니터링 비용을 낮출 수 있다. 증가된 효율성을 바탕으로 인터넷 기술은 비즈니스 요구 조건에 보다 신속히 대응할 수 있다.

위와 같은 효율성 제공을 위해, 우선적으로 그리드 컴퓨팅은 사용자가 원하는 데이터와 어떤 컴퓨터가 해당 요청을 처리하는지를 상세하게 알려줄 필요성이

*본 연구는 유비쿼터스 컴퓨팅 및 네트워크 원천기반기술과 과학재단 특정기초연구사업(R01-2004-000-10755-0)의 연구 결과로 수행되었음.

있다. 사용자가 원하는 서비스와 서비스 자체에 대한 기술 등에 대한 저장은 서비스 디스커버리(Service Discovery) 기술로 구현이 가능하다. PC 그리드 컴퓨팅에서의 텍스트 정보뿐만 아니라 컴퓨팅 파워, 데이터 저장 장치, 첨단 실험 장비 등이 어디에 있는지를 사용자에게 알려주는 디스커버리 기술은 SLP, JxTA, JADE, FIPAOS 등에 구현이 되어 있다[2,3,4,5]. 이러한 디스커버리 기술에 에이전트 플랫폼을 통합하여 그리드 컴퓨팅에서 보다 효율적인 자원 할당과 사용자가 원하는 데이터를 쉽게 찾을 수 있다. 에이전트 플랫폼은 각 학교나 기업에서 개발한 여러 가지 시스템이 존재한다. 이런 이기종 에이전트 플랫폼간의 호환을 위한 표준이 필요하다. FIPA 에서는 이기종 에이전트 플랫폼간의 호환 문제를 해결하기 위하여 디스커버리 미들웨어(Discovery Middleware)를 제안하였다[6]. 동일 에이전트 플랫폼 상의 에이전트는 자신의 데이터베이스를 이용하고, 서로 다른 에이전트 플랫폼 상의 서비스는 디스커버리 미들웨어를 통해서 해당 서비스의 위치나 세부 정보 등을 얻을 수 있다.

본 논문에서는 PC 그리드 시스템을 기반으로 각 FIPAOS 와 JADE 등과 같은 이종 에이전트 플랫폼 간의 서비스 디스커버리를 위한 디스커버리 미들웨어를 제안한다. 제안된 디스커버리 미들웨어를 통해서 사용자는 얻고자 하는 데이터와 어떤 컴퓨터가 요청을 처리하는지에 대한 정보를 효과적으로 획득할 수 있다. 본 논문의 구성은 다음과 같다. 2 장에서는 관련 연구로써 PC 그리드 컴퓨팅의 환경에 대해서 기술하고, SLP, FIPAOS 에 대해서 기술한다. 3 장에서는 FIPAOS 와 SLP 를 이용하여 PC 그리드 환경에서 서비스 디스커버리를 위한 디스커버리 미들웨어를 제안한다. 4 장에서는 시스템의 고려할 사항을 기술하고, 5 장에서는 결론 및 향후 연구 방향을 제시한다.

2. 관련 연구

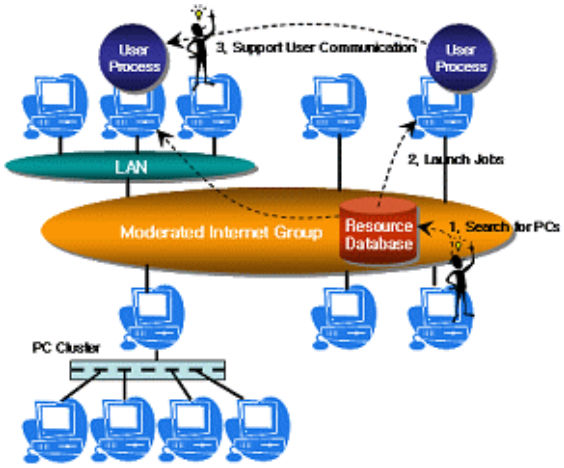
2.1 PC 그리드 컴퓨팅 환경

그리드 컴퓨팅이란 컴퓨터들을 네트워크에 연결하여 사용자가 원하는 네크워크의 컴퓨팅 자원이나 데이터 저장 장치 등을 필요한 때에 필요한 서비스로 사용한다는 개념이다. 이와 같은 개념에 따라 그리드 관련 기술과 서비스는 무척이나 다양한 형태를 취하고 있다. 그 중 하나인 PC 그리드 컴퓨팅은 일반 사무용 혹은 가정을 PC 들로 구성된 형태이다. PC 그리드 또한 두 종류로 분류된다.

첫 번째로 컴퓨팅 그리드(Computing Grid)이다. 컴퓨팅 그리드는 지역적으로 분산되어 있는 컴퓨팅 파워를 공유하여 마치 한 대의 고성능 컴퓨터처럼 사용할 수 있게 해 주는 것을 말한다. 두 번째로 액세스 그리드(Access Grid)이다. 액세스 그리드는 인간 상호작용의 지원을 목적으로 하는 프로젝트와 소프트웨어를 의미한다. 원격지의 연구자와 시각, 청각의 이미지를 공유하는 것에 의하여 고도의 공동 연구를 수행할 수 있다. 화면이나 카메라, 마이크의 수를 증가시킬 수 있기 때문에 액세스 그리드 노드를 임의의 규모로 구축할 수 있다. 그리고 분산된 지역의 연구원들이 공동으로 프로젝트를 진행할 수 있는 협업 환경을 제공한다.

2.1.1 PC 그리드 시스템 개략도

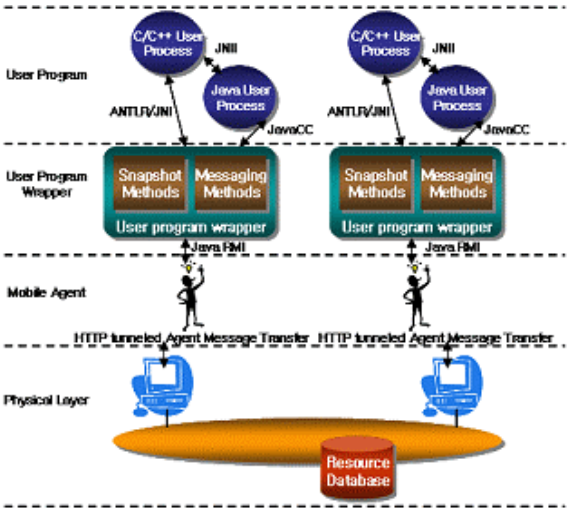
PC 그리드 개략도는 [그림 1]와 같다. 각 그룹 멤버의 정보를 데이터베이스에 유지하고 있는 인터넷 그룹에서 시작한다. 멀티플 인터넷 그룹은 글로버스 메타컴퓨팅 디렉토리 서비스 (Globus Meta-computing Directory Service)[8]를 사용해서 구성할 수 있다. 각각의 사용자는 게스트 계정을 생성하여 웹 서버에서 실행하며, 자신들의 컴퓨터 정보를 인터넷 그룹에 등록하고, 모바일 에이전트 실행 엔진은 컴퓨터 정보를 받게 된다. 사용자 작업과 관계된 모든 파일들은 모바일 에이전트를 통해서 전송이 된다.



[그림 1] PC 그리드의 개략도

2.1.3 4 계층 PC 그리드 환경

PC 그리드 시스템은 [그림 2]에와 같이 4 개의 계층으로 구성 된다[9].



[그림 2] 4 계층 PC 그리드 환경

물리적 계층은 네트워크에 연결된 클라이언트 노드들과 그룹 서버로 구성되며, 모바일 에이전트 계층은 클라이언트 사용자를 표현하고 작업 실행을 처리한다. 사용자 프로그램 wrapper 계층에서는 스냅샷 메소드와

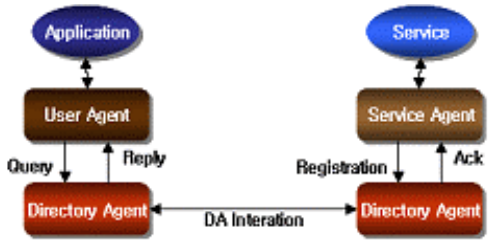
메시징 메소드를 제공하고 있다. 네 번째 계층인 사용자 프로그램은 Java 나 C/C++로써 쓰여진다

모바일 에이전트 기반 미들웨어 구현에 대해서 5 가지의 기술적으로 고려해야만 하는 문제들이 있다. 원격 컴퓨터에 모바일 에이전트를 배치 하는 방법, 프로세스 스냅샷을 가져오는 방법, 사용자 프로그램에서 전처리 하는 방법, 프로세스 중에서 통신을 용이하게 하는 방법, 5 가지의 기술적인 문제들이 있다.

2.2 SLP

SLP(Service Location Protocol)는 자바를 사용하여 만든 서비스 로케이션 프로토콜로서 서비스에 대하여 등록, 삭제, 수정 및 조회가 가능하다. SLP 는 실제 응용프로그램 형태의 서비스는 제공하지 않고 순수하게 서비스 디스커버리의 기능만을 제공하고 있다[2,7]. SLP 의 구조는 크게 DA(Directory Agent), UA(User Agent), 그리고 SA(Service Agent)로 구성되어있다. SA 는 특정 서비스를 제공하는 Provider 의 역할을 수행하는 모든 장치 또는 어플리케이션을 대표하며, UA 는 특정 서비스 사용을 요청하는 Consumer 라 할 수 있다. DA 는 SA 와 UA 사이에서 중재자 역할을 하며 서비스에 대한 모든 정보를 저장하고 있다. [그림 3]은 SLP 의 구조를 나타내고 있다. 먼저 UA 가 자신의 영역 내에 존재하는 DA 의 위치를 검색한다. 이때 멀티캐스팅 방식이 사용된다. 이에 DA 는 자신의 위치 정보를 UA 에게 넘겨주면, 다시 UA 는 원하는 서비스를 DA 에게 요청하게 된다. DA 는 자신의 데이터베이스에 해당 데이터가 존재한다면, 요청에 대한 SA 의 정보를 UA 에게 넘겨주게 되고, 이후로의 작업은 DA 를 거치지 않고 UA 와 SA 간의 유니캐스팅 방식으로 이루어 지게 된다. 하지만 이러한 서비스 디스커버리 기능은 SLP 에만 제한된 것으로 이기종 에이전트 플랫폼상의 서비스에 대한 조회가 불가능하다.

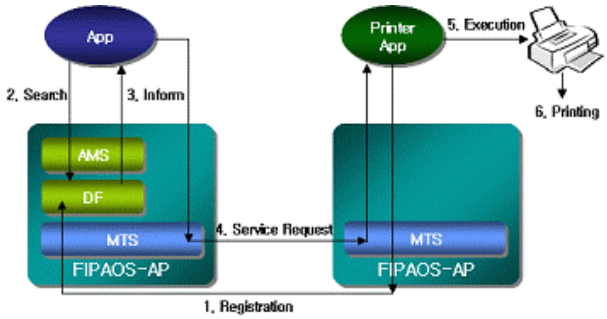
r



[그림 3] SLP 시스템 구조

2.3 FIPAOS

FIPA(The Foundation for Intelligent Physical Agents)는 에이전트들과 에이전트 기반의 응용프로그램들 간의 상호 운용을 지원하기 위한 개발 설계서를 공개함으로써, 지능형 에이전트 산업의 발전에 기여하는 국제 기구이다[8]. FIPAOS 는 이러한 국제 표준을 기반으로 만들어진 에이전트 플랫폼으로서, 서비스 디스커버리 기능을 포함함은 물론 실제 서비스를 제공한다. FIPAOS 의 구조는 [그림 4]와 같다. FIPAOS 에이전트 플랫폼은 크게 AMS(Agent Management Service), DF(Directory Facilitator), MTS(Message Transport Service)로 이루어져 있다[5]. 서비스 디스커버리 방식은 SLP 와 거의 유사하다. 먼저 서비스를 제공하는 에이전트가 서비스를 DF 에 등록시키고, 이를 원하는 사용자측의 에이전트가 DF 를 검색한 후 해당 서비스의 정보를 얻어온다. 그리고 나서 MTS 를 거쳐서 해당 서비스를 제공하는 에이전트와 통신하게 된다. 통신에 이용되는 프로토콜은 HTTP, IIOP, RMI 등이 있다.

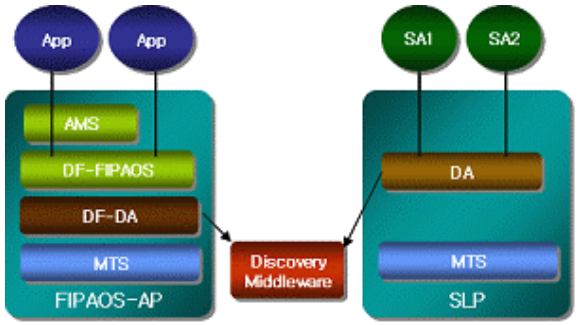


[그림 4] FIPAOS 시스템 구조

3. 에이전트 플랫폼 DM 모듈 개발

3.1 DM 모듈

본 논문에서 제안하는 FIPAOS 와 SLP 를 이용한 디스커버리 미들웨어(DM)의 구조는 [그림 5]와 같다.



[그림 5] SLP 를 이용한 DM 의 구조

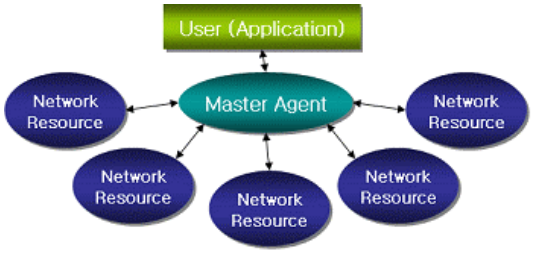
기본적인 구조는 FIPA 에서 제시한 ADS 와 DM 의 형태와 유사하다. 그러나 FIPA 에서는 서비스에 대한 정보를 각자의 에이전트 플랫폼 상의 데이터베이스에 담고 있기 대문에 서비스에 대한 요청이 있을 때 ADS 가 DM 과의 통신을 통해 질의를 보내게 되고, DM 이 다시 질의의 내용을 해당 에이전트 플랫폼에 전송하게 되는 즉, ADS ↔ DM ↔ AP 이러한 형식의 통신이 이루어지게 된다. 이렇게 될 경우 매번 서비스를 검색할 때마다 ADS ↔ DM ↔ AP 형식의 통신이 이루어 지기 때문에 네트워크 비용의 낭비를 초래할 수 있다. 이때 서비스가 존재하지 않을 경우에는 더욱 큰 낭비가 발생하게 된다. 따라서 본 논문에서는 [그림 5]와 같이 DM 과 관련된 에이전트 플랫폼 상의 데이터베이스에 저장된 서비스에 관한 정보를 FIPAOS 에이전트 플랫폼 상에 DF 의 형태로 저장시켜 놓음으로써, 매번 요청이 이루어질 때 마다 발생하는 네트워크 낭비를 예방하였다. 이렇게 되면 에이전트는 일단 자신의 DF 에서 서비스에 대한 검색을 하고, 매치되는 서 S 비스가 없을 시에는 바로 자신의 에이전트 플랫폼에 저장된 이종의 에이전트 플랫폼이 제공하는 서

비스를 검색하게 되어서 필요 없는 네트워크 비용의 낭비를 막을 수 있다.

3.2 Agent Grid

DM 모듈은 PC Grid 컴퓨팅과의 연동을 통해서 네트워크 자원들을 더 효율적으로 사용할 수 있다. 네트워크 자원은 크게 두 가지로 분류할 수 있다. 첫째는 하드웨어 자원으로 컴퓨터, 다른 장치, 병렬의 기계 등을 말한다. 두 번째로 소프트웨어 시스템이다. 그리드 운영 체제, 에이전트 소트프웨어 간의 통신, 보안 관리, 데이터베이스 등을 포함하고 있다.

[그림 6]에서는 그리드 컴퓨팅에 대한 통합된 에이전트 플랫폼을 보여주고 있다. 본 논문에서 제안한 에이전트 플랫폼을 사용함으로써 시스템 자원들의 관리를 제공한다. SLP 같은 프로토콜이나 FIPAOS 같은 이기종 간의 에이전트 플랫폼과의 상호 연동으로 사용자는 원하는 서비스를 보다 효율적으로 검색하고 획득할 수 있게 된다.



[그림 6] 그리드 컴퓨팅에 대한 통합된 에이전트 플랫폼

또한 에이전트 플랫폼에서의 마스터 에이전트들과 다수의 네트워크 노드에 존재하는 다양한 멀티플 에이전트 사이의 통신을 제공한다[9]. 에이전트들은 메타-컴퓨팅 툴킷들과 어플리케이션 레벨에서의 스케쥴링 메커니즘을 제공한다. 에이전트는 행동에 관련된 룰들을 포함할 수 있고, 다른 특정한 통신 언어를 사용해서 다른 에이전트들과 상호작용할 수 있다. FIPAOS 에서 통신에 사용된 HTTP, IIOP, RMI 와 같은 프로토콜을 이용함으로써 마스터 에이전트와 네트워크 노드에 있는 에이전트들 간의 상호 통신이 이루어질 수 있다. 각각의 에이전트들은 본 논문에서 제안한 DM 모듈을 통해서 다른 에이전트들이 가지고 있는 서비스에 대한 목록들을 제공 받을 수 있다. 서비스 목록을 바로 DM 모듈에 공시를 하여 각각의 에이전트가 제공하는 서비스는 무엇인지 그리고 상호작용할 수 있는 언어는 무엇인지를 기술할 수 있다. 다른 네트워크 노드에 있는 에이전트들의 모든 서비스들을 DM 모듈을 통해서 사용자는 서비스 목록을 제공받게 되고, 서비스 목록을 통해서 자신이 원하는 서비스가 어떤 것인지 더욱 효율적으로 서비스 받을 수 있다.

4. 고려사항

그리드 환경에서의 에이전트 플랫폼에 관련된 기술들은 매우 빠르게 개발되고 있다. 그러나 해결하고 고려해야만 하는 많은 기술적인 문제들이 아직 남아있다. 첫 번째로, 정확성과 성능에 관한 견고함을 들 수 있다. 두 번째로, 실시간 자원의 관리가 필요하겠다. 각 네트워크 노드에 있는 에이전트들이 가지고 있는 서비스가 실시간으로 유동적으로 변하는 그리드 컴퓨팅에서는 일정한 시간 주기마다 서비스 관련 정보들을 갱신하여, 가장 최신의 서비스를 사용자에게 제공하는 것이 필수이다. 세 번째로, 보안이다. 에이전트간의 통신이 이루어 질 경우에 보안 프로토콜이 필수적이다. 에이전트 보안 프로토콜은 키 분배 및 관리의 단순화, 인증 절차의 단순화, 에이전트 실행 플랫폼의 인증과 송수신 부인 방지 기능을 제공, 생명성을 보장, 실행 결과 데이터의 기밀성, 무결성을 보장해야 한다. 네 번째로, Fault-tolerance 에 대한 문제이다. 에이전트의 실행이 실패 했을 경우, 회복해야만 하기 때문에 Fault-tolerance 에 대한 문제도 고려해야 할 것이다.

5. 결론 및 향후 연구 방향

본 논문에서는 PC 그리드 컴퓨팅에서의 이종 에이전트들의 간의 상호 통신을 위해서 DM 모듈을 연구 개발하였다. 이기종간의 플랫폼에서의 서비스 디스커버리 기능이 필요하다. 디스커버리 기능을 본 논문에서 제안한 DM 모듈을 통해서 네트워크 노드들간의 서비스를 사용자에게 보다 효율적으로 제공하였다. 향후 연구로써는 에이전트의 보안 모듈을 첨가하여 사용자들이 신뢰성 있는 에이전트 플랫폼을 개발할 예정이다. 네트워크 자원을 더욱 효율적으로 활용하기 위해서 동시성 제어와 실시간 정보를 제공하는 방안과 에이전트의 Fault-tolerance 부분도 향후 연구해야 할 방안이다.

참고문헌

[1] Geoffrey Fox, “Grid Computing Environments” Copublished by the IEEE CS and the AIP, 20003
[2] James Kempf, Pete St. Pierre “Service Location Protocol for Enterprise Networks” Wiley Computer Publishing
[3] JxTA technology White Papers http://www.jxta.org/
[4] JADE Tutorial Giovanni Caire http://jade.tilab.com/
[5] FIPA-OS V2.2.0 Distribution Notes
[6] FIPA Agent Discovery Service Specification. Foundation for Intelligent Physical Agents, 2003 http://www.fipa.org/specs/fipa00095
[7] IPSJ Magazine Vol.44 No.6 June 2003
[8] M.Fukuda, Y. Tanaka, L. M. Campos, and S. Kobayashi.Programmability and performance of m++ self-migrating threads. In ***Proc. The IEEE Int’l Conference on Cluster Computing – Cluster2001***, Pages 331-340, Newport Beach, CA, October2001. IEEE-CS
[9] Zhihuan Zhang, Shuqing Wang “Agent-Based Framework for Grid Computing”, M.Li et al. : GCC 2003, LNCS 3032, pp. 629-632, 2004.