□ 研 究 論 文 □

韓國軍事運營分析學會誌
第 23 卷, 第 1 號, 1997 . 6 . 30

최상위 상호작용을 위한 효과적인 가상 네트워크 구축방법 (An Effective Way to Construction the Highest Interworkable Virtual Network)

석 진 원, 이 태 공*

Abstract

The network for providing information exchange function in distributed system is interconnected among other networks which are in the state of independent construction and operation. However, due to various network protocols and distributed applications, Problems occurred in the area of network intetconnectivity and interoperability of distributed applications in internetworking which should provide a good interoperability. This paper proposed an improved method for finding route which has the highest interworkability between end-systems using the interrelationships of network protocol. An algorithm and method of constructing a virtual network which is suitable for required distributed application are also proposed in this research.

* 국방대학원

1. 서 론

분산 시스템 환경 구축을 위하여 정보교환 기능을 제공하는 네트워크는 다른 네트워크와 상호연결되고 있다. 이러한 네트워크들의 상호연결에 의하여 분산응용(Distributed Application)의 상호운용이 이루어지며 원하는 정보를 획득할 수 있게 된다. 그러나 다양한 네트워크 프로토콜과 수많은 분산응용 및 제품들 때문에 네트워크간 상호연결 및 분산응용 상호운용성 보장문제가 제기된다.

본 논문에서는 네트워크 프로토콜에 의한 상호관계성 (Interrelat-ionship)을 이용하여 종단시스템간 상호운용성을 최대로 보장하는 경로를 식별하여 분산응용을 효율적으로 지원하고자 한다.

연구방법은 분산자원의 상호공유를 위하여 물리적 기기와 네트워크의 연결, 네트워크 상호간의 연결 및 분산응용의 상호연결을 위한 상호관계성인 연결성(Connectivity), 상호연결성(Interconnectivity), 상호작용성(Interworkability), 상호운용성(Interoperability)에 대해서 정의해보고, 이러한 상호관계성으로 네트워크 및 종단시스템간의 상호관계성을 식별하고, 분산응용을 원하는 종단시스템간 최상위 상호작용 가능한 가상 네트워크 구축방법 및 알고리즘을 제안한다.

2. 분산 시스템 상호연결

2.1 분산 시스템 개요

분산 컴퓨팅 시스템 (Distributed Computing System)이란 독립적인 많은 컴퓨터 장치들을 서로 연결하여 각 조직의 정보처리 욕구를 만족시켜주기 위한 시스템을 말하며, 분산 컴퓨팅(Distributed Computing)은 분산 컴퓨팅 시스템이 제공하는 각종 서비스를 말한다. 반면에, 중앙 집중식 시스템(Centralized System)은 컴퓨터 시스템을 구성하는 모든 요소들이 주장비(Host)를 중심으로 한 곳에 모여 있는 시스템을 말하며, 탈중앙 집중식 시스템(Decentralized System)은 시스템 구성요소들이 서로 제한적이거나 전혀 협조할 수 없도록 떨어져 있는 상태의 시스템을 말한다.

다시말하면, 시스템 구성요소들이 자율성(Autonomous)을 갖고, 어떤 전체적인 메커니즘에 의해서 작동되면서 상호협조가 되는 시스템을 분산 시스템(Distributed System)이라고 한다. 분산 컴퓨팅 시스템은 각각의 업무기능을 달성하기 위하여 네트워크를 통해 연결된 자율성 있는 컴퓨터의 집합체를 말한다[5].

2.2 분산 시스템 상호연결

분산 시스템을 구성하기 위한 고려사항들을 보면 다음과 같다.

첫째 어떻게 분산 시스템 환경에서 자원들을 상호연결할 것인가?, 둘째 어떻게 상호작용성을 보장할 수 있도록 네트워크를 연결할 것인가?, 셋째 어떻게 분산응용간에 상호운용성을 보장할 수 있도록 연결할 것인가 하는 측면에서 생각해 볼 수 있다. 그림1은 분산 시스템의 상호연결을 물리적 기기 상호연결, 네트워크 상호연결, 분산응용 상호연결의 관점 에서 나타내었다[3][5].

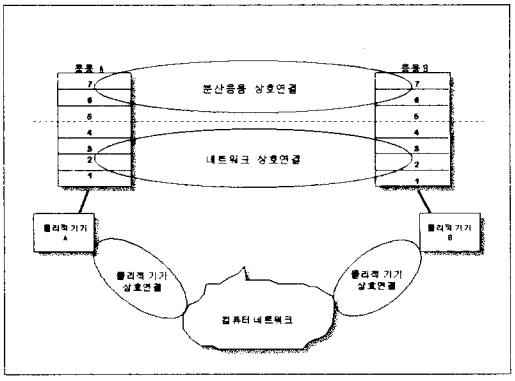

그림1. 분산 시스템 상호연결

2.2.1 물리적 기기 상호연결

물리적 기기 상호연결은 분산되어 있는 각종 기기(컴퓨터, 프린터, 각종 서버 등)들을 네트워크 어답터 또는 물리적인 케이블을 이용하여 네트워크에 연결시키는 것을 말한다. 이것은 OSI(Open System Interconnection) 참조모형의 물리계층과 데이터링크 계층에서 전송신호, 신호인터페이스, 메시지 형식 및 흐름제어 등은 주요한 기능이다.

예를 들면, 분산된 자원을 공유하려면 PC나 워크스테이션과 같은 기기 들은 네트워크에 연결됨으로써 메인프레임 등과 정보자원을 공유할 수 있는데, 모든 기기에 Ethernet 카드가 있을 때 Ethernet 케이블로 장치들을 물리적으로 연결하여 Ethernet을 구성할 수 있다. 이렇게 물리적 기기 들이 네트워크에 연결되는 것을 물리적 기기 상호연결이라 한다.

2.2.2 네트워크 상호연결

네트워크 상호연결은 분산된 각종 기기 들이 연결되어 있는 네트워크를 인터네트 워킹 장치(Relay)를 사용하여 다른 네트 워크와 연결하는 것으로 인터 네트 워킹(Internetworking) 이라 한다.

상호연결되는 네트워크로는 근거리(LAN), 중거리(MAN), 원거리(MAN) 네트워크 등을 들 수 있는데, 서로 떨어져 있는 네트워크들을 리피터(Repeater), 브릿지(Bridge),라우터(Router),게이트웨이(Gateway)와 같은 인터네트워킹 장치로 연결하여 정보를 상호 교환하게 하는 것을 말한다[10].

인터네트워킹 장치는 그림2와 같이 물리계층간의 네트워크 접속에는 리피터를, 데이터링크계층간에는 브릿지를, 네트워크계층간에는 라우터로, 전송계층 이상 또는 OSI 참조모형 전체 계층간의 접속에는 게이트웨이를 사용하며, 또한 브라우터(Brouter)는 브릿지와 라우터의 중간 기능을, 그라우터(Grouter)는 라우터와 게이트웨이의 중간 기능을 갖고 있다[6][7][10].

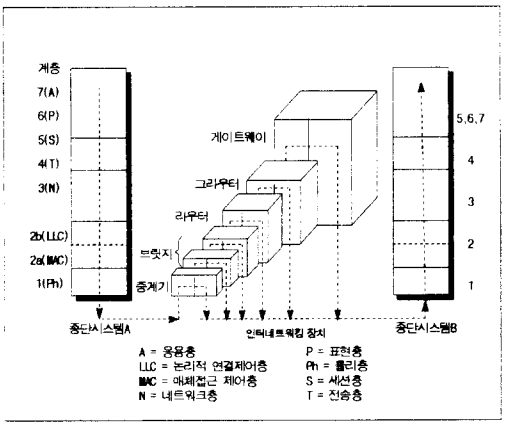

그림2. 인터네트워킹 장치(Relay)

먼저, 리피터는 일반적으로 LAN의 전송매체상에 흐르는 신호를 증형, 증폭, 중계하는 장치이다. 세그먼트간의 접속에 사용하는 것이외에 전송거리 연장, 배선의 자유도를 높이기 위해서 사용한다.

둘째, 브릿지는 데이터링크계층의 프레임을 중계하는 장치로 단말의 데이터링크층 주소(MAC 주소)를 식별해서 중계해야 할 프레임만을 통과시킨다. 브릿지의 중계기능은 일단 프레임을 브릿지 자체에서 읽고 주소를 해석해서 해당 세그멘트에 전송하는 스토어 앤드 포워드(Store and Forward) 방식을 취하는 장치이다.

셋째, 라우터는 네트워크계층 주소를 기초로 중계제어를 실현하는 인터네트워킹 장치로 송신지로부터 받은 패킷의 목적지 주소를 보고 최적경로를 선택하여 패킷을 중계한다.각 라우터가 이 중계 동작을 반복하므로써 데이터가 목적지까지 도달하게 된다. 라우터는 대규모 네트워크에서 주로 사용한다. 그 이유는 네트워크의 규모가 커지면 브릿지로는 단말노드의 매체접근제어(MAC) 주소를 유지하기 어렵기 때문이다.

넷째, 게이트웨이는 전송계층 이외의 상위계층이나 전체 계층간의 접속에 사용하는 인터네트워킹 장치로 일반적으로 응용계층 수준까지 다른 프로토콜로 동작하는 네트워크간의 접속장치를 말한다.

이와 같이 여러종류의 인터네트워킹 장치에 의해서 네트워크 상호연결이 성립되면, 각 종단시스템에 흩어져 있는 분산응용간의 상호운용성이 완전히 보장되어야만 종단시스템 사용자간에 분산응용 서비스가 가능해진다.

2.2.3 분산응용 상호연결

분산응용 상호연결은 종단 시스템에서 운용되는 응용들간에 서로 정보를 교환하려고 할 때 상호간의 분산응용이 정보를 교환하거나 공유하는 방법을 정의하는 것을 말한다.

분산응용인 단말기 에뮬레이션, 파일전송, 클라이언트/서버 상호동작 및 원격자료 접근 등의 분산응용을 지원하는 프로토콜에 의한 상위계층 상호운용성 문제는 매우 중요하다.

분산응용 서비스간에는 하위계층(계층1-4)의 프로토콜에 의해서도 상호운용성이 문제가 될 수 있으나, 상위계층(계층5-7)의 여러 분산응용의 프로토콜에 의해서도 지원하려고 하는 분산응용 상호운용성 보장문제가 제기될 수 있다.

최근 분산응용 상호연결에 있어 OSI 기반의 표준들은 다양한 종류의 분산응용에 이용되며 많은 표준 프로토콜들이 계속해서 개정되거나 향상되고 있다. 분산응용에서 자료를 전송하는 응용과 수신 응용간에 같은 프로토콜을 사용해야 상호운용성이 보장된다는 사실은 모두 알고 있는 사실이다.

이와 같이 네트워크간의 상호연결에서 인터네트워킹 장치로 연결된 네트워크상의 분산응용 상호간의 소통을 위하여 상호운용성을 보장하는 연결이 필수적이다. 예를 들면, 분산응용의 한 형태인 터미널 기능수행을 위하여, 만약 두 종단시스템간에 사용되는 응용이 OSI 참조모형에서 정의한 표준 프로토콜인 VT를 사용한다면 상호운용성 문제는 없을 것이다. 그러나 만약 한쪽에서 VT(Virtual Terminal)를 사용하고 다른 쪽에서는 TCP/IP 프로토콜인 Telnet을 사용한다면 서로의 상호운용성을 위해서는 변환기나 게이트웨이와 같은 인터네트워킹 장치를 사용해야 한다[5].

지금까지 설명한 물리적 기기 상호연결, 네트워크 상호연결, 분산응용 상호연결이 보장된다면 종단시스템간에는 완전한 상호운용성이 보장되어 사용자는 투명한(Transparency) 서비스를 지원받을 수 있을 것이다. 그러나 물리적 장치에서부터 네트워크간 상호연결 보장, 분산 응용서비스 지원에 이르기까지 시스템 전체의 명확한 구조에 의한 상호운용성 보장은 매우 어려울 뿐만아니라 네트워크간에 어느 정도의 상호운용이 가능한지 식별하는 것 조차도 쉽지 않다. 이와 같이 분산응용의 상호운용성 보장은 네트워크를 사용하여 자원을 공유하려는 한 계속적으로 고려되어야 할 것이다.

그러므로 분산응용의 상호운용성 보장을 위하여 각 네트워크의 프로토콜을 이용한 상호관계성(연결성, 상호연결성, 상호운용성, 상호작용성)을 정의한다. 정의된 상호관계성은 최상위 상호작용 가능한 가상 네트워크를 구성하는 기반을 이룬다.

3. 상 호 관 계 성

3.1 연결성

분산환경에서 상호관계성은 크게 직접연결 형태와 간접연결 형태로 분류할 수 있다. 직접연결 형태는 집단화 관계와 종료관계를 기반으로한 연결성(Connectivity Relationship)이 있고, 간접연결 형태는 연결성을 기초로 만들어지는 상호관계로 상호연결성(Interconne-ctivity Relationship), 상호작용성(Interworkability Relationship), 상호운용성(Interoperability Relationship)이 있다[11].

그림4의 연결성 관계에서 (가)의 집단화 관계(Aggregation Relationship)는 객체클래스에서 "~의 일부분이다(is-a-part-of)"라는 관계로 나타낸다.

즉, "네트워크장치는 네트워크의 일부분이다 : 네트워크는 네트워크 장치를 일부분으로 가진다 (has-as-a-part). 또는 연결(Link)은 네트워크의 일부분이다 : 네트워크는 연결을 일부분으로 가진다"와 같은 관계를 집단화 관계라 한다.

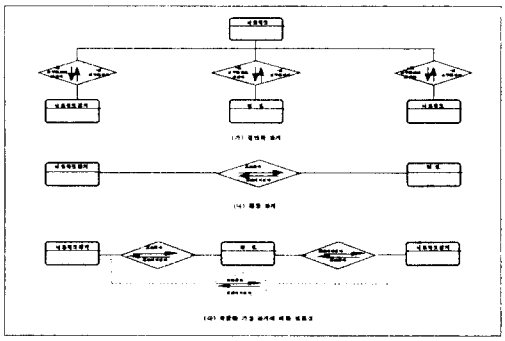

그림4. 연결성 관계

(나)의 종료 관계(Termination Relationship)는 연결성 문제의 기준이 되는데 어떤 연결이 네트워크장치와 연결된 마지막 부분일 때 "네트워크장치는 연결에서 종료한다(terminate) : 연결은 네트워크장치에서 종료되어진다(is terminated)"와 같은 상호관계성으로 나타낸다.

(다)의 연관된 가상관계에 의한 연결성 형태는 네트워크 장치간의 연결에 의해서 종료관계로 연결되어 질 때 두 네트워크 장치간에는 종료관계에 의해서 연결성이 보장되므로 가상적으로 연결성이 존재

한다고 본다.

이와 같이 세가지 연결성 형태를 기반으로한 직접 연결 형태인 연결성은 두 인접노드가 직접 연결되어 있을 때를 말하며, 임무쌍은 "연결한다(connected-to)"와 "연결되어진다(is-connected-to)"이다. 이 관계가 존재할 때 연결성이 존재한다고 한다. 일반적으로 연결성은 두 네트워크가 연결되었으면 존재한다고 본다.

3.2 상호연결성

상호연결성은 인접노드간 연결성 체인(chain)으로 임무쌍은 "상호연결한다(interconnected-to)", "상호연결되어진다(is-interconnected-to)"이다. 그림 5 에 상호연결성 형태를 나타낸다. 상호연결성은 아래 세가지 형태로 유추할 수 있다. (가)의 함축적 가상관계(Implicate Virtual Relationship)는 A가 B와 연결되었으면 A와 B는 함축적으로 상호연결되어있다고 한다. (나)의 분산적 가상관계(Distributive Virtual Relationship)는 A가 B와 상호연결되어 있고, B가 C와 연결되어 있으면, A와 C는 분산적으로 상호연결되었다고 한다. (다)의 이행적 가상관계(Transitive Virtual Relationship)는 A가 B와 상호연결되어 있고, B와 C도 상호연결되어 있을때, A와 C는 이행적으로 상호연결되었다"고 한다[3][11].

3.3 상호운용성

상호운용성은 OSI 참조모형에서 특정 계층을 기준으로 정의되는 데 예를 들면, "네트워크장치 A와 B는 전송계층에서 상호운용성이 있다"고 정의할 수 있다. 상호운용성을 위한 임무쌍은 "n-계층에서 상호운용한다"와 "n-계층에서 상호운용되어 진다"이다. 여기서 n은 관계되는 OSI 참조모형의 계층들을 범위로 하는 메타변수이다. 예를 들면, 네트워크 계층간에 상호운용성이 있다면 임무쌍은 "네트워크 계층에서 상호운용한다"와 "네트워크계층에서 상호운용되어 진다"로 표현한다. 그러므로 n-계층을 가진 참조모형에 대해서 n-계층의 다른 가능한 상호운용성의 임무쌍이 있다. 그러나 모든 상호운용성이 이행(Transitive) 가능하지는 않다.

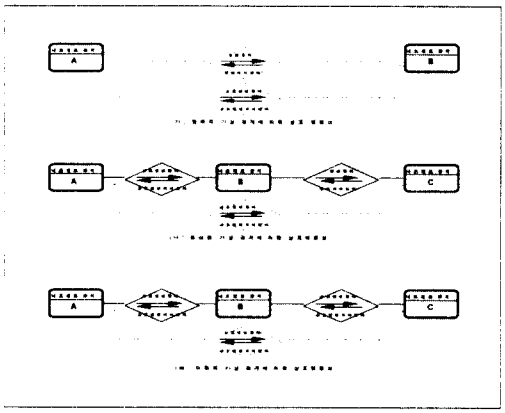

그림5 상호연결성 관계

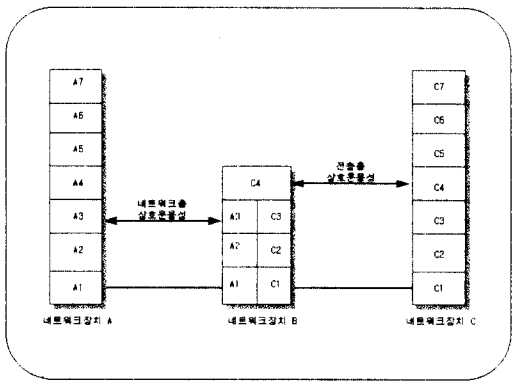

그림6. 프로토콜 스택의 쌍매칭에 의한 상호운용성 유도

그림6에서 프로토콜 스택의 쌍매칭에 의한 상호운용성 유도에서 네트워크장치 A가 네트워크장치 B와 서로 상호운용성이 있고, 네트워크장치 B가 또한 네트워크장치 C와 상호운용성이 있을지라도, 네트워크장치 A가 네트워크장치 C와 상호운용 가능하다고 할 수 없다. 이것은 네트워크장치 B가 두 개의 완전히 다른 프로토콜 스택들의 집합에서 동작할 수 있기 때문이다. 상호운용성은 정보의 상호교환을 위하여 필요조건은 아니다. 이것은 변환기(Translator) 또는 게이트웨이에 의해 비록 쌍방간에 상호운용성이 존재하지 않을지라도 데이터를 교환할 수 있기 때문이다.

3.4 상호작용성

먼저, 네트워크장치 A와 B가 서로 파일전송을 할려고 할 때 네트워크장치 A는 OSI 표준 프로토콜인 FTAM(File Transfer, Access, Management)을 사용하고, 네트워크장치 B는 TCP/IP의 프로토콜인 FTP(File Transfer Protocol)을 사용한다면, 네트워크장치 A와 B는 OSI 참조모형의 응용계층에서 상호작용성이 존재하지 않지만 인터네트워킹 장치로써 변환기를 사용한다면 상호간에 파일을 전송할 수 있을 것이다. 이 경우 상호운용성은 네트워크장치 A와 변환기 그리고 변환기와 네트워크장치 B간에 존재한다. 그림7은 프로토콜 변환기에 의한 상호작용성의 유도를 나타낸다.

상호작용성은, "두개의 네트워크장치가 서로 상호운용성이 특정 OSI 계층에 존재하지 않지만 인터네트워킹 장치인 변환기나 게이트웨이를 통해 자료를 교환할 수 있을 때 상호작용성이 있다"고 한다. 상호작용성은 상호작용성 관계의 이행성과 상호운용성에 의한 상호작용성 분포로 발견할 수 있다.

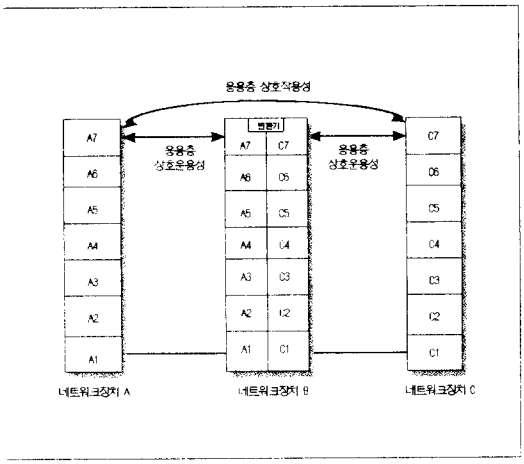

그림7 .프로토콜 변환기에 의한 상호작용성 유도

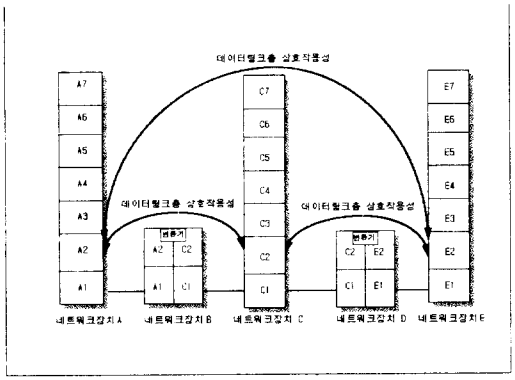

그림8. 상호작용성 관계의 이행성

첫번째, 그림8은 상호작용성 관계의 이행성을 나타낸다. 그림에서 "네트워크장치 A는 네트워크장치 C와 데이터링크층에서 상호작용성이 있고, 네트워크

장치 C는 네트워크장치 E와 데이터링크층에서 상호작용성이 있다"면 "네트워크장치 A는 네트워크장치 E와 데이터링크층에서 상호작용성이 있다"라고 한다. 그러나 상호운용성에서는 이와 같은 이행성이 항상 성립되지 않는다.

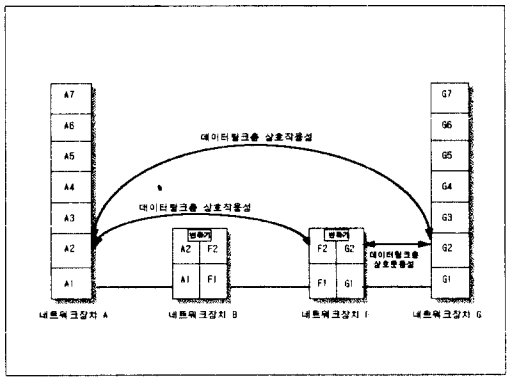

그림9. 상호운용성에 의한 상호작용성 분포

두번째, 그림9는 상호운용성에 의한 상호작용성의 분포를 나타내는데, "네트워크장치 A는 네트워크장치 F와 데이터링크계층에서 상호작용성이 존재하고, 네트워크장치 F는 네트워크장치 G와 데이터링크계층에서 상호운용성이 있으며, 네트워크장치 F가 데이터링크계층에서 변환기능을 가진다"고 하면 "네트워크장치 A는 네트워크장치 G와 데이터링크계층에서 상호작용성이 존재한다"고 한다.

3.5 다중 연결에 의한 상호 운용성 식별

앞에서 설명된 상호관계성에 의해서 두 종단시스템간의 상호작용성이 결정되면, 다음으로 지원하려는 분산응용을 위해서 두 종단시스템간 식별된 상호작용성의 상위계층에 대해서 상호운용성을 식별하여야 한다. 식별된 상호운용성에 의해서 기존의 네트워크가 초기의 지원없이 서비스를 지원할 수 있는지 여부가 결정가능하다. 지원이 불가능할 때는 적당한 인터네트워킹 장치에 의해서 지원 요구된 서비스를 위해서 네트워크를 상호연결해 주어야 한다.

이렇게 서비스를 원하는 두 종단시스템간에 여러 중계시스템이 있어 이들이 상호운용성 및 상호작용성에 의해서 서로 연결되고, 최종적으로 종단시스템간의 상위 프로토콜에 의해서 상호운용성이 식별될 수 있다. 이를 다중 연결에 의한 상호운용성이라 한다. 언급된 각종 상호관계성을 정리해보면 그림10과 같이 나타낼 수 있다.

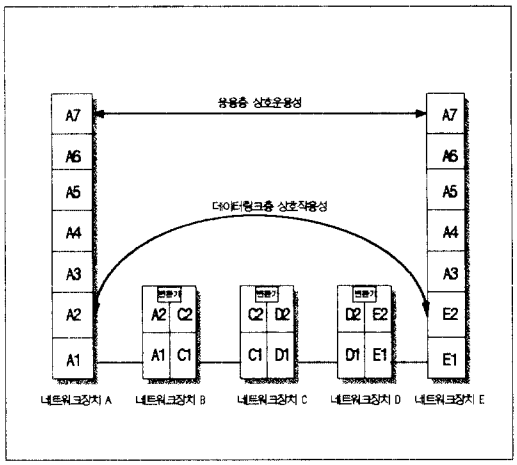

그림10. 다중연결에 의한 상호운용성 유도

첫째, 다른 네트워크장치간 연결성은 어느 네트워크장치가 동일한 연결객체에 의해서 종료관계를 갖는지 결정함으로써 알 수 있다.

둘째, 다른 네트워크장치와 상호연결성은 연결성 체인에 의해서 알 수 있다. 즉, 함축적 가상관계, 분

산적 이행관계, 이행적 가상관계에 의한 상호연결성으로 알 수 있다.

셋째, n계층 상호운용성은 n-계층까지 동일한 서브스택이나, 네트워크장치는 하나의 프로토콜 스택을 일부분으로 가지고 있다는 연결성 관계가 존재하면 알 수 있다.

넷째, n-계층 상호작용성은 다음 세가지로 유추할 수 있다.

가. n-계층에서 " 공통 네트워크장치가 하나의 프로토콜 변환 객체를 일부분으로 가지고 있다"면 n-계층에서상호운용이 존재 한다고 볼 수 있다.

나. n-계층에서 변환 객체의 조건이 되는 n-계층 호 운용성에 대한 분포에서 알 수 있다.

다. n-계층의 상호작용성의 이행성으로 부터 알 수 있다.

다섯째, (n+1)-계층의 상호운용성은 그림10에서 (n+1)-계층에서 동일한 프로토콜 객체를 갖는 네트워크장치와 하위계층에서의 n-계층 상호작용성에 의해서 알 수 있다.

4. 최상위 상호작용 가능한 가상 네트워크 구축

4.1 네트워크 상호연결 그래프

최상위 상호작용 가능한 가상 네트워크 구축에서 그래프를 이용하여 네트워크간의 상호연결을 표현한다. 그래프(Graph)는 점과 선으로 구성된 집합으로 우리가 어떤 문제를 해결하려고 할 때 자주 사용하게 된다. 그래프는 시스템의 구조 및 상황 등을 표현하는 것이 매우 자연스럽고 표현이 간단해서 많이 사용된다[1][2][3].

제안하는 최상위 상호작용 가능한 가상 네트워크 구축 알고리즘에서는 네트워크간의 상호연결 형태를 그래프를 사용하여 나타내는데, 각 종단시스템이 연결되어 있는 네트워크를 그래프의 노드(Node)로, 네트워크간의 상호연결을 간선(Edge)으로 표현한다. 표현된 그래프를 네트워크 상호 연결 그래프(Network Interconnection Graph)라고 정의한다. 본 논문에서는 네트워크 상호연결 그래프를 다음과 같이 정의한다.

[정의 4.1] : 네트워크 상호연결 그래프

네트워크 상호연결 그래프(이하:NIG)는 종단시스템과 중계시스템(또는 중계 네트워크)을 개체(Entity)로 표현하고, 개체간의 상호연결 관계를 간선으로 표현한 양방향성 그래프이다. 또한 정의된 그래프는 노드와 노드간의 연결관계를 상호관계성으로 나타내어 가중치를 부여한 비용(Cost) 그래프이다.

[정의 4.2] : 노드(Node)

노드는 각각의 네트워크를 나타내며 그래프에서 원으로 표현한다. 원내부에는 노드 식별자를 표기한다. 또한 노드는 네트워크의 프로토콜 정보를 가지고 있고 인접노드와의 상호관계성표를 유지관리한다.

[정의 4.3] : 간선(Edge)

간선은 실선으로 표현하며 네트워크간에 양방향성을 가지므로 표현된 간선은 양방향성을 갖고 있다고 간주한다. 간선에는 노드간의 상호관계성을 가중치로 부여하여 나타낸다. 가중치 범위를 본 논문에서는 비연결에서부터 계층별로 0에서 7까지로 부여한다.

위에서 정의된 NIG를 이용하여 네트워크에 연결된 종단시스템간에 분산응용을 지원하고자 할 때, 각 네트워크나 종단시스템이 보유하고 있는 프로토콜을 이용하여 최상위 상호운용성을 보장할 수 있도록 상호관계성을 식별하여 가상 네트워크를 구축할 수 있도록 하는데 그래프의 최단경로 탐색방법을 적용할 수 있다.

정의된 NIG의 시작노드와 목적노드사이에서 분산응용을 지원하고자 할 때 여러개의 중계노드를 거쳐서 목적노드에 도달하게 되는데, 이때 시작노드와 목적노드 및 각 중계노드가 가지고 있는 프로토콜의 상이성이 우리가 운용하려는 분산응용의 상호운용을 저해하게 된다. 이를 위하여 분산응용을 원하는 종단시스템간에 최상위 상호운용성 식별방법을 모색하고자 한다.

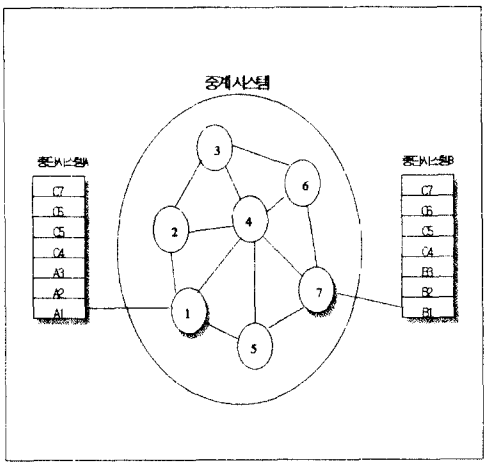

그림11. 분산응용을 위한 네트워크 예

예를 들면 그림11은 분산응용을 위한 네트워크의 종단시스템A에서 중계시스템을 통하여 종단시스템B와 전자우편(X.400)을 주고 받고자할 때, 두 종단시스템간에는 이를 처리할 수 있도록 상호운용성이 보장되는 프로토콜이 탑재되어야 하고 경로상의 중계시스템들은 원하는 정보를 교환할 수 있도록 중계해 주어야 할 것이다.

여기서 종단시스템A와 종단시스템B간에 프로토콜 상호운용성을 찾기위하여 탐색알고리즘을 적용해 종단시스템B까지 찾아갈 수 있다.

그러나 효율적인 상호운용성 보장을 위해서 종단시스템간에 같은 상호작용성을 보장하는 모든 경로를 찾고, 이를 가상 네트워크로 구성해서 신뢰성있는 정보의 전송을 보장할 수 있도록 해야한다.

4.2 최상위 상호작용 가능한 가상 네트워크 구축과정

인터네트워크에서 분산응용을 지원하기 위한 가상 네트워크 구축방법으로, 먼저 네트워크를 NIG로 나타내어 각 노드와 인접노드의 상호관계성을 식별하고 식별된 계층에 따라 가중치를 부여한 후, 각 노드의 상호관계성표를 구성하고 이를 이용하여 목적노드간에 최상위 상호작용성 및 이를 지원하는 가상 네트워크를 구축하고자 한다.

효율적인 분산응용 지원을 위하여 최상위 상호작용 가능한 네트워크 구축과정은 그림12와 같이 나타낼 수 있다.

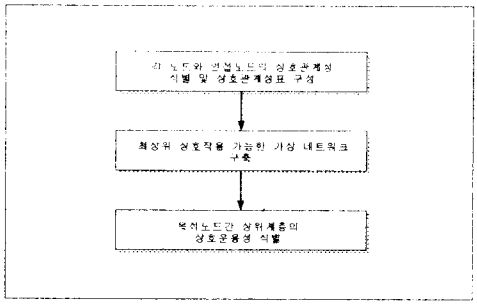

그림12. 최상위 상호작용 가능한 가상 네트워크 구축과정

4.2.1 상호관계성 식별 및 상호관계성표 구성

제안된 최상위 상호작용가능한 가상 네트워크 구축의 각 단계를 설명해보면, 먼저, 각 노드가 인접노드와 상호관계성 정도에따라 상호관계성표를 구성하여 각 노드에서 유지하게 하는 단계이다.

먼저 NIG의 어느 한 노드와 인접노드의 상호관계성을 계층별로 식별하고 가중치를 부여하여 상호관계성표를 구성한다. 구성되는 상호관계성표에는 인접노드명과 노드간의 상호관계성 정도에 따라서 비연결에서부터 계층1, ..., 계층7까지 가중치를 0에서 7까지 부여한다. 정의된 가중치는 표1과 같다.

상호관계성 식별 계층	가중치
비연결	0
물리 계층	1
데이터링크 계층	2
네트워크 계층	3
전송 계층	4
세션 계층	5
표현 계층	6
응용 계층	7

표1. 상호관계성에 따른 가중치

표1의 가중치를 각 노드의 상호관계성 식별시 적용하여 각 노드에 연결된 인접노드의 상호관계성표를 구성할 수 있는데 그 예는 표2와 같다.

인접 노드명	가중치
2	5
3	2
4	3
.	.
.	.
m	6

표2. 상호관계성표 예

```
PROCEDURE ConstructInterrelationshipTable
  input : current_node(i), adjacency_node(j)
  output : interrelationship_table IT[j]
   /* 현재노드(i)로부터 인접노드(j)까지 프로토콜 계층 1에 대하여 */
   /* 상호관계성표 구성 */
  FOR all adjacendy_node(j) of current_node(i) = 1, n DO
  BEGIN
    int i, j, n, weight.
     /* 현재노드(i)로부터 인접노드(j)까지 연결성 식별 */
    IF node i to node j = chain of connectivity relationship
    THEN  node i and node j = interconnectivity relationship
    /* 현재노드(i)로부터 인접노드(j)까지 상호연결성 식별 */
      WHILE (not top of protocol layer or exist interoperability or exist
              interworkability) DO
         /* 현재노드(i)로부터 인접노드(j)까지 프로토콜 계층(n)에 대하여 */
         /* 상호운용성과 상호작용성 식별 */
        BEGIN
           /* 현재노드(i)로부터 인접노드(j)까지 상호운용성 식별 */
          IF node i to node j = chain of layer(n) interoperability
                                relationship
          THEN node i and node j = layer(n) interworkability relationship
           /* 현재노드(i)로부터 인접노드(j)까지 상호작용성 식별 */
          ENDIF;
          layer n:=layer(n+1);
           /* 상위계층의 상호관계성 파악을 위하여 pop-up */
      ENDWHILE;
      weight:=layer(n); /* 식별된 상호관계성 */
    ELSE weight:=0; /* 비연결에 의한 가중치 설정 */
    ENDIF;
     IT[j]:=interrelationship(node j, weight)
     /* 식별된 상호관계성을 각 노드의 상호관계성표에 저장 */
  ENDFOR;
END;
```

알고리즘1. 상호관계성표 구성 알고리즘

각 노드에서 상호관계성표를 구성하는 것은 상호작용성 식별을 위한 관련정보를 제공하기 위한 것으로써 이러한 정보는 노드의 추가, 삭제, 프로토콜의

변경이 발생하였을 때 각 노드에서 항상 새로운 정보에 의해서 상호관계성표를 재구성해 주어야 한다. 각 노드와 인접 노드의 상호운용성을 식별하기 위한 상호관계성 표 구성은 알고리즘1 ConstructInterrelationshipTable 로 타내었다[3][11].

4.2.2 최상위 상호작용 가능한 가상 네트워크 형성

다음 단계는 각 노드의 상호관계성표에 의하여 분산 응용을 원하는 목적노드간 최상위 상호관계성을 보장하는 모든 경로를 식별하고, 가상 네트워크를 구축하는 단계로써 식별된 경로에서 각 노드간에 최저의 상호작용성이 최상위 상호작용성이 된다. 제안하는 알고리즘은 최단경로 탐색 알고리즘을 응용하여 각 노드간의 상호작용성을 탐색해 나간다. 이때 최단경로 알고리즘은 최소비용의 합으로 경로를 선정해 나가지만 제안된 알고리즘은 각 노드간의 상호관계성이 가장 큰 경로를 임의의 한 노드와 인접한 모든 노드에 대하여 최대최소화 연산을 수행하여 최상위 상호작용성을 구한다. 식별된 최상위 상호작용성을 이용하여 목적노드간 최상위 상호작용 가능한 모든 경로를 찾아낼 수 있다. 제안된 알고리즘은 세 단계로 나타낼 수 있다. 첫째, 각 노드정보를 입력하여 노드간 최상위 상호작용성을 식별하는 단계이다. 둘째, 식별된 최상위 상호작용성을 가지고 목적노드간 최상위 상호작용성을 제공하는 모든 경로를 찾는 단계이다. 셋째, 식별된 경로에 의해서 최상위 상호작용 가능한 가상 네트워크로 형성하는 단계로 나누어 볼 수 있다. 다음 알고리즘2가 최상위 가상 네트워크 구축 알고리즘이다[3][4][11].

```
PROCEDURE ConstructVirtualNetwork
input:Weight_Table WT;Start_Node SN;Destination_Node DN;
output:Highest_Interworkability IW;Path_table Path[n];Visited[n];
       Virtual_Network;
/* 단계 1 : 모든 노드간에 각 노드의 상호관계성표를 이용하여 최상위 */
/* 상호작용성을 식별한다 */
BEGIN
  int sn, dn, i, j, k, ptr, n, m, l;
  Boolean T[i];
  IF (SN≠DN) THEN
     FOR (k=1 TO n) DO
                FOR (i=1 TO n) DO
                   FOR (j=1 TO n) DO
                       IF (T[i]≠true) THEN
                         W(i, k):=MAX(WT(i, k), MIN[WT(i, k), WT(k, j)]);
                      /* 모든 노드에 대하여 최대최소화 연산을 수행하여 */
                       /* 최상위 상호작용성을 식별한다 */
                         ENDIF;
                   ENDFOR;
                  T[i]:=true; /* 방문된 노드기록 */
                ENDFOR;
 ENDFOR;
 ENDIF;
 IW:=W(sn, dn); /* 두 목적노드간에 식별된 최상위 상호작용성 */
 /* 단계 2 : 두 목적노드간에 최상위 상호작용성을 보장하는 모든 */
 /* 경로를 식별한다 */
 IF (sn=dn or W(sn, dn)=0) THEN
      Print "Input Error!!!";
      RETURN; /* 두 목적노드가 같거나 비연결시 종료한다 */
 ptr:=1;Path(ptr):=dn;is:=-1;js:=1; REPEAT DO
LABEL:EXIT;
   REPEAT DO
    FOR i:= is, n DO
     IF WT(i, dn) ≥ IW THEN
               FOR j=js, k DO
                 IF Highlink=IW AND i=sn THEN /* 시작노드 여부 판단 */
                   dn:=dn+1;
                   FOR l=ptr, 1, -1 DO
                       PRINT Path(ptr); /* 식별된 경로를 출력한다 */
                   ENDFOR;
                 ELSEIF Highlink<IW THEN js:=1;
                 ELSEIF Highlink>IW THEN
                   FOR l=1, ptr DO
                              IF i=Path(ptr) THEN js:=1;GOTO EXIT;
                   ENDFOR;
                   ptr:=ptr+1:Path(ptr):=i;
                   Visited(ptr):=j;  /* 방문노드기록 */
                   dn:=i;IW:=W(dn, j);is:=-1;js:=1;
                 ENDIF;
                 GOTO EXIT;
               ENDFOR;
     ELSE js:=1;
     ENDIF;
    ENDFOR;
    is:=Path(ptr);js:=Visited(ptr);ptr:=ptr-1;dn:=Path(ptr);
    IW:=MIN[W(is, js), WT(is, dn)];
    IF js=k THEN is:=is+1;js:=1;
           ELSE js:=js+1;
    ENDIF;
  UNTIL ptr:=1;
  /* 단계 3 : 식별된 경로를 이용하여 최상위 상호작용 가능한 가상 */
  /* 네트워크를 구성한다 */
  Construct Virtual_Network by selected The_Highest_Interworkability
          Paths;
  Print Virtual_Network; /* 구성된 가상 네트워크를 출력한다 */

END;
```

알고리즘2. 최상위상호작용 가능한 가상 네트워크 구축 알고리즘

4.2.3 목적노드간 상위계층의 상호운용성 식별

마지막 단계는 전단계에서 구축된 최상위 상호작용 가능한 가상 네트워크에 의해서 지원할 목적노드간 분산응용을 위하여 식별된 최상위 상호작용성의 상위계층에 대하여 목적노드간 상호운용성을 식별하는 단계이다. 식별과정은 상호관계성표 구성과 같이 노드간 상호운용성의 임무쌍을 식별하여 결정한다. 이 단계가 끝나면 목적노드간에 요구된 분산응용의 지원여부를 알 수 있다. 상위계층 상호운용성 식별 알고리즘은 알고리즘3에 나타내었다.

```
PROCEDURE FindHighLayerInteroperability
  input:start_node(i);destination_node(j)
  output:ResultInteroperability
   /* 목적노드간 최상위 상호작용성의 상위계층인 계층(n+1)에 대하여 */
   /* 상호운용성 식별한다 */
  IF (layer(n) interworkability and same protocol stack at layer(n+1))
  THEN node(i) and node(j)=layer(n+1) interoperability
  ENDIF;
  WHILE ((not top of protoco layer) and (same protocol layer)) DO
   BEGIN
     IF (not matching protocol start_node(i) and destination_node(j)
         at layer k)
     THEN Exist Interoperability at layer k
      /* n+2≤k≤top of protocolstack */
      /* 프로토콜 스택을 pop-up하면서 계층7까지 상호운용성 식별한다 */
     ENDIF;
  ENDWHILE;
  ResultInteroperability:=k; /* 식별된 목적노드간 상호운용성 */
END;
```

알고리즘3. 목적노드간 상위층 상호운용성 식별 알고리즘

4.3 최상위 상호작용 가능한 가상 네트워크 구축 예

앞의 그림11의 예에서 종단시스템A와 종단시스템B가 전자우편(X.400)을 주고받고자 할 때, 앞절에서 제안한 최상위 상호작용 가능한 가상 네트워크 구축 방법을 이용하여 단계적으로 최상위 상호작용성을 보장하는 가상 네트워크를 구축해보면 다음과 같다.

4.3.1 [단계 1] : 상호관계성 식별 및 상호 관계성표 구성

정의한 NIG를 이용하여 네트워크를 표현하고 각 노드간 상호관계성을 식별하여 상호관계성표를 그림13과 같이 구성할 수 있다. 각 노드의 상호관계성표는 알고리즘1에 의해서 각 노드와 인접노드에 대하여 상호관계성표를 구성한다. 생성된 상호관계성표는 각 노드의 상호관계성표에서 유지하는 것으로 가정한다. 상호관계성표는 네트워크 확장시나 변동사항이 있을때 알고리즘1에 의해서 재구성된다.

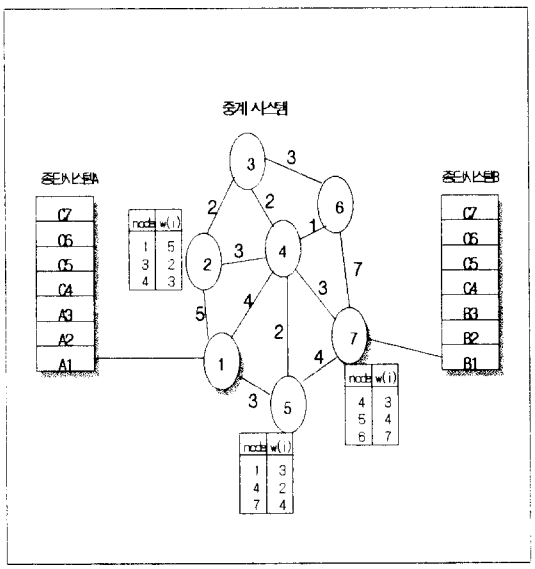

그림13. 상호관계성표 구성결과

4.3.2 [단계 2] : 최상위 상호작용 가능한 가상 네트워크 형성

제안된 알고리즘2를 이용하여 [단계 1]에서 구성한 각 노드의상호관계성표를 종단시스템A는 노드1에 접속되어 있으므로 시작노드는 1로, 종단시스템B는 노드2에 접속되어 있으므로 목적노드는 7로하여 상호작용성을 최대로 보장해주는 경로를 찾아 식별하면 {1(5)->2(3)->4(3)->7}의 경로이고, 최상위 상호작용성은 경로의 최저 가중치인 '3'이 된다. 여기서 ( )안은 각 노드사이의 가중치를 나타낸다.

다음으로 최상위 상호작용성 '3'을 보장 하는 경로를 모두 식별해 보면 {1(5)->2(3)->4(3)->7,1(4)->4(3)->7, 1(5)->5(4)->7}과 같은 세 개의 경로를 식별할 수 있다. 식별된 경로를 최상위 상호작용 가능한 가상 네트워크로 형성하면 그림14와 같다.

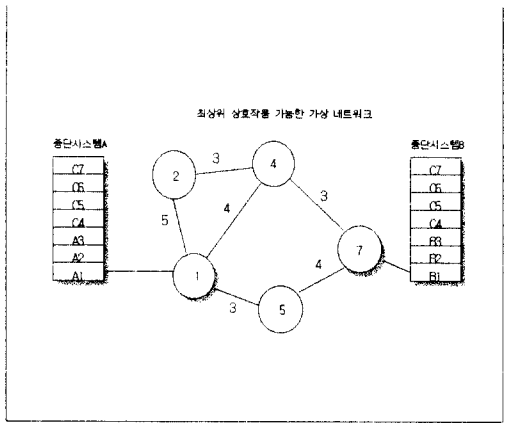

그림14. 구축된 최상위 상호작용 가능한 가상 네트워크

4.3.3 [단계 3] : 목적노드간 상위계층의 상호운용성 식별

식별된 상호작용성의 상위계층에 대하여 노드1과 노드7에서 상호운용성을 식별하는 단계로 종단시스템A와 종단시스템B는 상호작용성이 '3'이므로 계층3 이상의 상위계층에 대한 상호운용성을 알고리즘3을 적용하여 식별한다.

두 종단시스템의 상위계층 상호운용성이 계층4에서 계층7까지 모두 같은 프로토콜을 사용하므로 상호운용성은 계층7까지 존재한다. 그러므로 두 종단시스템간에는 요구된 분산응용인 전자우편 서비스(X.400)를 부가사항없이 지원가능하다고 판단할 수 있다. 그림15에서 목적노드간 상위계층 상호운용성 식별결과와 분산응용의 지원여부를나타내었다.

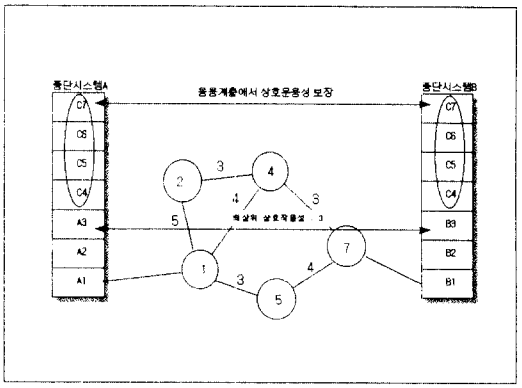

그림15. 목적노드간 상위 계층 상호운용성 식별결과

5. 결 론

본 논문에서는 네트워크 프로토콜에 의한 상호관계성을 이용하여 분산응용을 운용하려는 종단시스템간 상호작용성을 최대로 보장하는 모든 경로를 찾아서 요구된 분산응용을 효율적으로 지원할 수 있는

최상위 상호작용 가능한 가상 네트워크 구축방법 및 알고리즘을 제안하였다. 제안된 가상 네트워크의 구축 과정은 다음과 같다.첫번째, 각 네트워크의 프로토콜을 이용하여 네트워크간 상호 관계성을 식별하여 각 노드의 상호 관계성표를 구성한다.
두번째, 분산응용 서비스를 원하는 종단 시스템간에 제안된 알고리즘을 사용하여 최상위 상호작용 경로를 모두 탐색하여 가상 네트워크를 구축한다.
세번째, 목적 노드간 요구된 분산응용의 지원 여부를 결정하는 단계로 이와같이 세 개의 단계에 의하여 가상 네트워크를 구축할 수 있도록 제안 하였다.

제안된 최상위 상호작용 가능한 가상 네트워크 구축방법의 적용에 의해 기대할 수 있는 효과로는 요구된 분산응용을 네트워크가 지원할 수 있는지 알아볼 수 있고, 인터네트워킹 장치의 선정단계에 적용하여 상호운용성을 보장할 수 있을 것이다. 마지막으로 제안된 알고리즘은 중앙집중 형태로써 분산 알고리즘 형태에 대한 연구와 네트워크상에서 효과적으로 적용할 수 있는 방법에 대한 연구가 있어야 할 것 이다.

참고 문헌

[1] 강맹규,네트워크와 알고리듬, 박영사, 1991.

[2] 김병수," K-최대용량 계산법에 관한 연구", 국방대학원 석사논문, 1989.

[3] 이태공,"인터네트워킹(Internetworking)을 위한 효과적인 릴레이(Relay) 선정 방법",국방대학원, 1995.

[4] 주경숙, 최종원, " 분산기법을 사용하여 최대용량 경로를 찾는 새로운 접근방법", '94 가을 정보과학회 학술발표 논문집B 제21권 2호, 1994.

[5] Amjad Umar,Distributed Computing and Client -Server Systems , Prentice Hall, 1993.

[6] Colin Smythe, Internetworking: Designing the Right Architecture, Addison-Wesley, 1995.

[7] R. J. Cypser, Communications for Cooperating Systems: OSI, SNA and TC-P/IP ,The system programming press , 1991.

[8] Ellis Horowitz , Strataj Shani , Fundamentals of Datastructures in Pascal,Computer Science Press. 1991.

[9] Gerard Tel, Distributed Algorithms, Cambridge university press, 1994.

[10] Stalling, Data and Computer Communications, Prentice Hall, 1994.

[11] Subodh Bapat, Object-Oriented Networks, Prentice Hall, 1994.