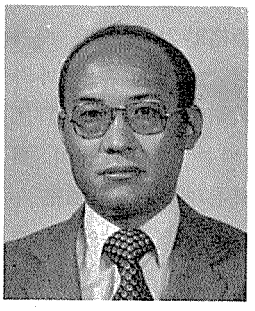
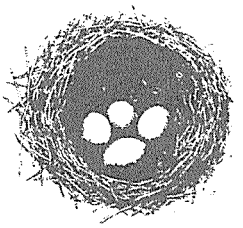


吳 吉 祿
韓國電子通信研究所
컴퓨터연구부장 / 工博

분산 컴퓨터 시스템



I. 서 론

분산 시스템이란 무엇인가? 단일 컴퓨터 자체도 CPU에서 보면 registor logic과 multiplier logic이 서로 분리되어 상호 수행한다는 관점에서 본다면 분산되었다고 할 수 있으므로, 모든 컴퓨터 시스템은 무조건 분산 시스템이라고 할 수도 있겠다. 그것이 마땅치 않다면, 통신 network에 의해 연결된 computer들이 상호 협력 수행하는 시스템이 분산 시스템이라고 보는 것이 온당하다고 볼 수 있을지도 모른다. 현재까지는 완전히 일반화된 정의는 없으며, 그 특성 및 범위 정도를 말하면, 분산 시스템이란 여러 시스템과 user process들이 message에 의한 통신방법을 사용하여 서로 상호 협력 수행하는 컴퓨터 시스템이라고 말할 수 있는 정도이겠다.

즉 여러 interactive한 component들의 구성이다. 그러면 이러한 분산 시스템이 왜 근래에 연구개발이 활발한가? 그 이유는 크게 두가지로 볼 수 있다. 첫째는 microelectronics기술과 통신기술의 발전이며 둘째는 사용조건에 맞추어 다양한 컴퓨터 종류가 개발되었으며 사용자들은 이러한 것들을 경제적인 면으로, 계산능력 향상이라는 면으로 또는 사용상 편리하다는 이유로 분산 시스템을 요구하게 됐다는 것이다. 통계를 보면 프로세서, 메모리 등의 가격은 약 5년에 10분의 1 정도로 떨어지고 있고 앞으로도 계속 그러할것 같다. 이로 인해 컴퓨터의 종류가 다양해지고 사용분포도 바뀌었는데 예를 들면 1975년도에 mainframe : 83%, mini : 10%, 기타 7% 이던 것이 1985년도에는 PC : 22%, SBS : 13%, WP : 10%, mini : 21%, mainframe : 34%로 사용 분포가 다양해졌다.

그리고 초기에는 단순히 중앙 컴퓨터에 멀리 떨어져서 Terminal을 연결하고자 하는 목적으로 시작한 데이터 통신기술이 근래에는 packet

switching 기술 그리고 보다 근래에는 소위 Local Area Network이라는 것이 매우 빠른 속도로 발전되어 짧은 거리부터 먼거리까지 서로 컴퓨터들을 연결시키는 기술이 그 특성은 서로 다르나 다양하게 발전되었다고 볼 수 있다. 이들 발전된 기술을 바탕으로 점차 일이 처리되는 근원지 또는 information이 필요 또는 발생되는 근원지에 컴퓨터를 위치시켜 놓는 것이 보다 경제적이 되었으며 그것이 가능하여졌다. 컴퓨터들을 접합(coupling) 시키기가 용이하여졌으므로, 이를 바탕으로 만들 수 있는 장점인 처리능력 향상, 안정성 및 신뢰도 향상, 자원공유 등을 사용자에게 제공할 수 있게 되었다.

그러나 근본적으로 분산 시스템이란 어떤 process도 전체 real system state를 알 수 있는 순간이 없으므로 인해 발생되는 문제들이 많고 그 해결이 복잡하므로 이러한 분산 시스템의 문제점 및 제한점을 해결하는 control 기술의 설계발전이 중요 열쇠라고 하겠다. 왜냐하면 분산 시스템은 단순히 비집중(Decentralized) 되었다는 의미는 아니고 어떤 원칙과 규칙을 가지면서 조직력으로 구성되어져 상호 협력해야만 그 의미가 있고 장점들을 발휘시킬 수가 있다는 것이다. 실제로 보더라도 현재까지 연구되어지거나 개발되는 분산 시스템을 보면 완전하게 Program, data 그리고 Control을 분산 시스템의 각 component에 분산시킨 시스템은 아직 없다.

Ⅱ. 분산 시스템의 종류와 목적

분산 시스템의 종류를 여러 관점 즉 접합도에 따른 구분, 접속매체에 따른 구분, 연결 구조형태에 따른 구분, 각 component 간의 interdependancy에 따른 구분, 그리고 운영체제 형태에 따른 구분 등으로 나누어 볼 수 있겠으나, 여기서는 접속매체와 접합도에 따라 구분하여 본다. 여러Component를 어떤 형태건 또는 어떤 연결 수단을 가지고서든지 연결시켜 상호 협력 수행하는 것이 분산 시스템이긴 하지만 연결시키는 수단이 제공하는 성격에 따라서 구성되어지는 분산 시스템의 양상은 많은 영향을 받게 된다. 즉 전달거리, 전송속도, 전송 지연시간 등이 process 간 통신형태에 많은 영향을 미치며 또한 Component 들의 접합도의 강약을 결정지을 수 있다. 여기서 전송 지연시간의 장단과 불규칙성이 미치는 영향도 큼을 간과하지 말아야 한다.

왜냐하면 첫째, 어떤 순간에 있어서도 한 machine의 어떤 process도 다른 어떤 machine의 개략적인 state 밖에 알 수 없게 되기 때문이며 (공통 clock을 사용치 않고 delay가 길고 변화하여 두관측 순간이 일치할 수 없음) 둘째는, message가 수신된 순서와 그것이 송신됐던 순서가 틀릴 수가 있기 때문이다. 우선 먼거리에 걸쳐 분산된 시스템, 즉 LHN(Long Haul Network)에 의한 분산 시스템이 있다. LHN의 특성을 보면 거리는 10km이상, 전송속도는 300bps ~100bps 정도이며 Network상의 Error rate는 크며 전송 지연시간도 크며 매우 불규칙하다.

그리고 message전달 비용은 message size와 Error Rate에 비례하게 된다. [그림 1]에서 보듯이 전송속도 및 거리의 범위는 일반적으로 그러하다는 것이지 반드시 그 범위안에 있으라는 것은 아니다. 즉, 먼거리에서도 높은 전송속도 및 낮은 Error Rate 등을 갖출 수 있겠지만 통신비가 너무커서 일반화시키기는 어렵다. 미국의 Arpanet, 불란서의 Transpac, 한국의 SDN, Dacomnet 등이 있다. LHN의 여건하에서는 Component들은 서로 Weakly coupled, 또는 Loosely coupled의 약한 접합도를 갖게 된다. Remote Terminal들이 Modem, Multiplexer 또는Concentrator 등을 사용하여 re-

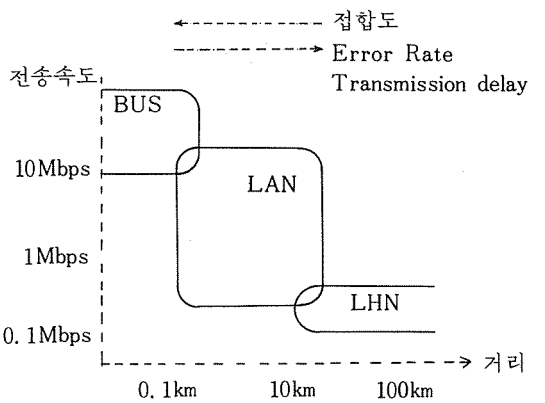


그림 1

mote access가 가능케 되어 비행기 예약이나 Banking system 등의 응용 분야를 갖게 할 수 있다. 또는 컴퓨터들이 서로 연결되어(주로 Packet Switching network으로의 경향) receiver를 sharing 하며 분산처리(Distributed Processing) 시스템을 구성하게 된다.

다음에는 LAN(Local Area Network)에 의한 분산 시스템을 들 수 있겠다. LAN의 성격도 거리는 0.1km~10km, 전송속도는 0.1Mbps~10Mbps, Error rate는 낮고 전송 지연시간은 비교적 짧고 고른 편이다. message 전송 비율은 message size에 비례하게 된다. LAN의 여건하에서는 loosely coupled, strongly coupled, 또는 closely coupled의 접합도를 갖는 것이 가능케된다. LAN은 어느 정도 지역적인 분산도를 가지면서 전송속도 등의 장점을 고루 갖추므로 Component 들은 서로 중간 정도의 접합도를 갖는 것이 알맞을 것이다. 그런데 LHN의 Software를 LHN에 설치하거나(예 : Unix 4.2 bsd) 또는 shared memory 환경에서의 software 구조 형태를 LAN에 Adaptation하여 설치하는(예 : Star O. S., Medusa) 경우가 있다.

따라서 LAN 환경에 맞는 중간 정도의 접합도를 Component들 사이에 제공하는 Software에 관한 연구 활동이 활발하며 ETRI의 DADOS가 그 범주에 속한다. LAN을 이용하여서 일반적으로 일부 component들은 각기 고유의 특수 서비스 기능을 갖고 나머지는 그 서비스를 공유하며 User task를 수행하는 형태의 조직으로써 구성되는 server system 형태가 있을 수 있으며, Independent 한 component 들이 연결되어 전체적으로는 user에게 마치 한 시스템인 것처럼 동작할 수 있게 하는 Symmetrical(Integrated) System 형태가 있을 수 있다. 전자는 개발에 드는 노력이 적게 들 수 있으나 Functional Redundancy의 부족 즉 service 기능을 갖는 한 Component의 fail이 전 시스템의 fail을 가져올 수 있는 단점이 있다. 세번째로 Bus 또는 Interconnection Network에 의한 분산 시스템을 생각할 수 있다. 그 특성을 보면 거리는 100m 이내, 전송속도는 10Mbps 이상, Error rate는 매우 낮고 Transmission delay는 매우 짧고 고르며, message 전송 비용 또한 매우 낮다.

이러한 특성을 이용하여 Very Strongly Coupled, tightly coupled, closely coupled의 다양한 그리고 비교적 강한 접합도를 갖는 분산 시스템을 구성할 수 있다.

그런데 일반적으로 shared memory를 공용 통신 수단으로써 사용하는 tightly coupled system은 분산 시스템의 특성 및 범위에서 제한되는 경향이 있으므로 여기에서는 일단 다루지 않기로 한다. 어쨌든 짧은 거리에서 매우 높은 전송 속도와 낮은 Error rate 등을 보장하므로 Processing Element들이 연결되어 System Performance의 증가를 꾀하는 목적이 추가되며 또는 Data Flow 구조의 컴퓨터를 구성하는데 사용되기도 한다. 기존 병렬선의 Bus에 의한 연결들이 Noise 문제, 선간의 전송지연 차이의 문제, 거리의 제한 등의 단점이 있으므로 이를 초고속 LAN으로 대치해서 연결하여 System Performance를 올리려하는 시도도 발견된다.

먼거리를 매우 높은 전송속도 및 용량 그리고 매우 큰 전송지연 및 원천적으로 broadcast 형태의 통신특성을 지니는 위성통신에 의한 component들의 연결이 특이하다. Data security 등의 문제가 있지마는 고유의 장점으로 인해 이를 이용한 분산 시스템의 발전이 예상된다. Singgle processor computer에서도 Virtual하게 분산이라는 관점을 갖을 수 있는 데, 예를 들면 process들이 각기 independent하게 특수한 task를 수행하며 주어진 IPC 방식에 의해 상호 협력 수행하는 경우를 말한다.

이는 시스템 설계 단순화의 잇점과 시스템 각 부분 사이의 단순한 interaction의 장점 즉 module화의 장점을 가져올 수 있다. 이상에서 살펴본 바와 같이 분산 시스템의 종류는 매우 다양하므로, 그 목적 또한 각기 다를 수가 있다. 그래서 분산 시스템에게 요구되어지는 일반적인 목적을 살펴본다.

우선 performance의 증대이다. shared memory에 의한 기존의 multiprocessor들은 bottleneck과 contention의 문제를 내포하므로 decentralized 된 control 기술에 의하여 각기 processing element들이 상호 협력하는 방향으로 performance 증대를 이룰 수 있다. 문제는 con-

trol의 분산이 많은 기술적 난제를 안고 있다는 것이다.

둘째는 확장성이다. 시스템에서 performance의 증감이 요구되거나 또는 어떤 서비스의 기능이 추가 또는 변경되는 것이 필요할때 분산 시스템에서는 Dynamic하게 변경 확장 가능하다는 것이다. 즉 분산 시스템에서는 modular 한 구조를 갖을 수 있으며 따라서 system설계의 단순화, 설치 및 maintenance의 간편화의 장점을 가져올 수 있다.

세째는 availability의 증가이다. H/W 적으로는 여러 모듈을 중복 설치하고 S/W 적으로는 fault, error, failure를 발견, 회복시키는 기능을 갖게 하여 안정성을 제공할 수 있다. physical system에서 전혀 사고가 발생치 않을 수는 없으므로 분산 시스템에 의하여 시스템 전체의 안전성을 높이는 것이 중요한 목적이 된다.

네째는 자원의 공유이다. 자원이라 함은 기기, service, data 등이 될 수 있으며 이를 여러 process 들이 공유함으로써 경제성과 시스템의 효율을 높일 수 있다. 이를 위하여는 각 자원에 process들이 access할 수 있는 기능을 제공하는 것도 중요하지만 자원의 할당이 시스템 전반에 걸쳐 적합하게 이루어지도록 하는 control 기능이 시스템의 효율을 위해 중요하다.

그리고 기타로는 통신비 절감, 지역적 분산, 높은 reliability 등을 목적으로 생각할 수 있으나 이러한 목적들은 서로 연관되는 의미들이다.

Ⅲ. 분산 시스템의 구조

분산시스템의 component를 크게 그 구조를 나누어 보면 H/W, O.S, Application program의 세부분으로 구성된다고 볼 수 있다. 여기서 component를 연결시키는 통신의 기능이 세부분의 어느 곳에서 일어나느냐에 따라 분산시스템의 특성 및 구조가 분류되어질 수 있다. LHN에서 Heterogenous computer들에 의한 분산시스템의 경우를 보면 message 전달을 하기 위한 Common 표준이 O.S. level에서 있을 수 없으므로 통신기능이 있는 부분은 각기의 Local Operating System (LOS) 위에 분포하게 된다. 따라서 접합도는 약하게 되고 서로 각 component의 access하는 object의 범위가 매우 제한되게 된다〔그림 2〕.

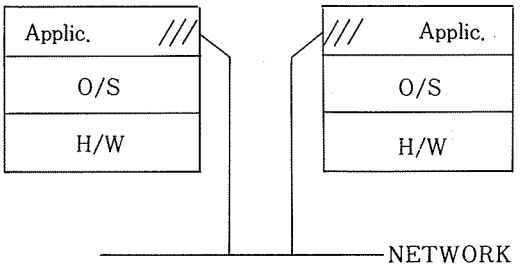


그림 2

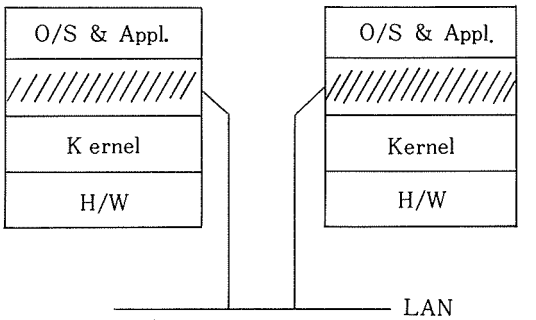


그림 3

접합도가 강하여지려면 통신기능이 있는 부분은 아래로 내려가게 되며, 서로 access할 수 있는 object의 범위가 넓어지고, 따라서 process간 상호 협력하는 범위도 넓어지게 된다. 〔그림 3〕과 같이 Newcastle Connection의 경우를 보면 user process에서의 system call 들이 통신 layer에서 intercept되어 local 또는 remote site의 kernel 기능을 Access할 수 있게 되며, 이때 Network는 user로부터 감추어지면서 Access Transparency를 제공하게 된다. ACCENT, CHORUS, ETRI의 DADOS 등LAN에 의한 중간 접합도의 시스템들을 보면 통신기능이 kernel level에 존재하게 된다〔그림 4〕.

사용자에게 access transparency, location transparency 등을 제공하여 분산시스템이라는 것을 user로부터 감추어 줄 수 있게 된다. 이런 경우 분산운영체제는 homogenious해야 하며, shared memory일때 보다는 정보 공유의 정도는 낮고 공유하는 Data의 granuality는 크게 된다. Server System에 의한 분산시스템의 경우를 예로 들면 V-kernel과 같은 경우 그 구조

가 [그림 5]와 같다.

여기서는 통신기능 부분이 보다 아래에 위치하게 되는데, 이는 shared memory 환경에서의 S/W 구조 형태를 LAN에 적용 설치하는 경우로 볼 수 있겠다. Network이 완전히 갖추어져 uniform한 user 환경을 제공할 수 있으며 또한 개발에 드는 노력이 비교적 적게 드는 반면 functional redundancy의 부족이 단점이 된다.

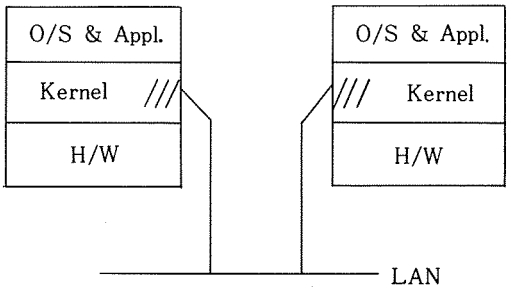


그림 4

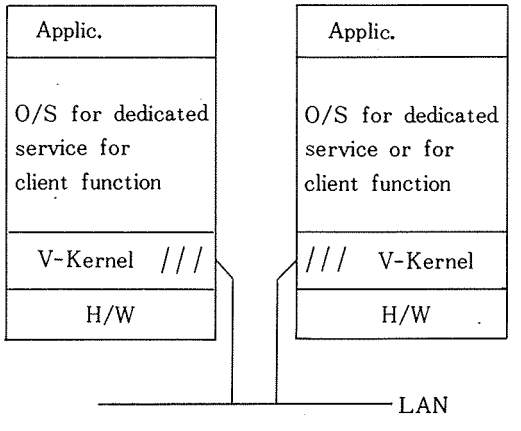


그림 5

Ⅳ. 통신기술

초기의 컴퓨터에서는 punched card 를 입력수단으로 사용하는 즉 off line으로 사용자가 컴퓨터를 사용하였으나, multiprogramming 방식에 의해 CPU를 time sharing 하게 됨에 따라 terminal들을 통해 interactive하게 컴퓨터를 사용하게 되었고 modem을 사용하여 terminal이 지역적으로 분산될 수 있게 되었다. 여러 terminal을 한 곳에 붙일 경우 경비 절감을 위하여 multiplexer, concentrator가 나왔고, Host processing overhead를 줄이기 위하여 Front End Processor가 개발되었다. 이러한 것들은 주로 대중 전화선을 사용하여 circuit switching 으로 통신이 이루어졌다.

그런데, local에서도 컴퓨터가 설치, 활용되어감에 따라 컴퓨터간의 통신이 필요하여졌고, 이들이 서로 전용선들로 연결됨에 따라 message switching이 유효하여졌고 이어 그 다음엔 보다 효율적인 packet switching에 의한 Network이 발전하게 되었다. Packet switching 기술은 미국의 Arpanet에서 처음으로 채택하여졌고 이어 Cyclade, Tymnet 등에서 채택되었다. 그러다가 통신 network이 대중 통신매체로서 발전하면서 부가되어진 기능들: Error detection, Retransmission, Error recovery 등등의 기능들이 추가되어, 일반대중들은 부가가치된 선로에 연결만 하면 되는 형태의 Public service Data network이 나오게 되면서 또 하나의 전기가 마련되게 되었다.

예로는 Telnet, Datapac, Dacomnet 등이 있으며 여기서는 앞서보다 line 사용 효율, Reliability 등이 높아지고 국제표준 즉 X.25 등이 등장되는 시기므로 그 채택이 가능하여졌다. 장래에는 더욱 더 부가가치되는 즉 error 검출 수정기능 강화, Data service 기능 강화, transmission line의 reliability 증가, 다양한 Transport service 제공 등의 발전이 있으리라 판단된다. 통신 시스템들은 보다 module한 형태로 바뀌게 될 것이며 장래에는 Terminal들이 자체적으로 X.25기능을 갖추게 될 전망이다.

또한, Telephone (Voice), Telegraphy, Data transmission의 서비스 기능이 단일 network 시스템에 의해 제공될 것이다. Computer based device가 많이 개발 설치되어 사용됨에 따라 LAN의 사용이 활발하여졌다. LAN은 일반적으로 Data PABX, baseband LAN 등의 형태를 가지고 있으며 환경에 따라 선택되어 사용되고 있는데 High speed LAN에 의해 Voice / Image / Data 전송이 복합 제공될 전망이나, LAN에 의한 분산시스템의 개발이 활기를 띄고 있으며 Bus에 의해 Module들을 연결하였던 것들이 LAN에 의해 대치될 전망이다. Process

간의 통신형태는 일반적으로 1:1이 대부분이었으나 1:N, N:1, 또는 N:N 통신이 발전되어 group 개념에 의한 communication이 발전되어 분산시스템의 효율이 증대하게 될 것이다.

어쨌든 통신이라 하면 실체들(entities)이 서로 정보교환을 수행하는 것을 말한다. 이를 수행하려면 정보 교환을 주고받는 두 entity 사이에 이들이 접근 가능한 공동 object가 존재해야 하고, 이 object는 Channel이라 불리운다. Channel은 memory, line, fiberoptic 등의 형태위에 구성되며 이들은 여러 실체들이 공유해야 (multiplexy Contention, reservation 등) 효율적이 될 것이며 이렇게 형성된 Channel logical channel이라 한다. 이러한 channel들도 실체들이 접근할 수 있도록 묘사되어져야 하며 이를 interface라 하고 또한 이 channel을 통하여 실체들이 필요한 어떤 기능을 수행키 위하여 서로 수행상의 규칙이 필요하여지는데 이를 Protocol이라 한다. 통신실체들을 규정하기 위해서는 여러 level 즉 layer를 갖는 것이 단순화 및 open system으로 하기에 효율적이다.

그리하여 ISO에서는 OSI reference model이 7 layer 구조를 갖고 있고 이것이 세계적으로 받아들여지고 있는 형편이다. 간단히 살펴본 바와 같이 통신기술의 발전은 channel의 성능 강화, 방식의 발전, interface & protocol의 발전에 의해 이룩되어진다는 것을 알 수 있다.

V. 분산시스템 소프트웨어 기술

분산시스템 S/W System이란 여러 processor들이 통신 네트워크로 연결된 형태의 구조에서 수행되어져야 할 S/W들을 말하며 일반적으로 여러 layer 구조로써 설계되어진다. 핵심부분인 분산 운영 체제는 여러 형태로 구분할 수 있겠지만, 일반적으로는 크게 두 부류 즉 Network operating system(NOS), Distributed Operating System(DOS)로 나눈다. NOS에서는 Host가 각기 independent 한 Local Operating System(LOS)을 가지고 그 위에 통신 management 기능이 부가되어 서로 통신 및 resource naming을 하게 된다. 이때 시스템의 optimization은 각기 LOS에서 행하여진다.

따라서 heterogenous 시스템에 의한 분산 시스템 구성에 좋으나 시스템간의 제한된 정도의 cooperation만이 가능해진다. 예로는 NSW, RSEXEC, XNOS 등이 있으며 LHN 같은데서 많이 볼 수 있다. DOS에서는 단 한 종류의 O/S가 모든 분산 component에 올라가게 되며 따라서 분산시스템의 모든 resource들은 coherent하게 묶을 수 있게 된다. 예로는 RIG, APOLLO, ACCENT, ROSCOE, TRIX 등이 있으며 LAN, 또는 BOS의 환경에서도 많이 볼 수 있다. DOS 중에서 data와 control까지 비집중화 된 것들을 distributed processing operating system(DPOS)라고 하는데 완전한 형태의 DPOS는 많은 장점을 주지만 현재 완전한 DPOS는 개발되지 못하고 있다.

예로는 CHOROS, MEDUS, MICRO OS, STAR OS 등이 있다. 분산시스템 운영체제를 user 관점에서 자세하게 분류하여 보면 다음과 같다.

첫째, Autonomous system들의 구성이라는 형태인데 이것은 각기 LOS를 갖고 있는 형태인데 system을 user로부터 감추기 위한 어떤 기능 즉 transparency 제공기능도 없는 것이다. 예로는 UUCP, FTP 등이 서로 다른 운영체제 하에서 운영되는 것을 들 수 있다.

둘째로는 Autonomous system들의 구성이며 Network은 감추어진 형태인데 이것은 Local이나 Remote이나 같은 access 형태를 갖지만 (Access Transparency) 그 location에 관한 정보는 그 local name space에 가지고 있어야 한다.

예로는 NSW, Newcastle connection, Cocanet 등이 있다.

세째로는, Autonomous system 들이 집적화되어 마치 한 시스템처럼 보이게 해주는 것인데 이때는 location control 등에 관련된 정보가 모든 user에게 같은 name space에 있게 된다. Homogenous 환경 즉, DOS 형태의 것인데 이때는 user에게 Access Transparency, location transparency 그리고 control transparency, execution transparency(Process Migration에 의한 Load Balancing) 까지도 제공할 수 있게 된다. 예로는 PULSE, MOS, CHORUS, DADOS 등이 있겠다.

마지막으로는 세째와 같이 Autonomous 하지 못한 시스템들이 집적된 시스템인데 이는 주로 server system 형태에서 볼 수 있으며, 특수 component 들에 어떤 주어진 서비스 기능이 주어지게 되며 다른 component들은 user task를 수행하게 된다. Redundancy의 부족이 단점이며 예로는 Arachm, Cambridge Distributed Computing system, V-kernel 등이 있다.

세번째와 네번째는 주로 LAN 환경에서만이 설계, 개발되는 경향이다. Integrated 분산시스템의 중요한 한 단계로서 분산 File system 이 있는데 분산 File system은 user에게 마치 하나의 Logical File system이 있는 것처럼 보여주는 것이 목적이 된다. 예로는 DFS, LOCUS등이 있으며 directory assignment replication 등이 주요 연구과제가 된다. 분산 DBMS (Database Management System) 또한 개념적으로는 마치 database system이 centralized 된 것처럼 보여주어야 하는데 DDBMS가 가지는 장점은 Reliability, Availability, 빠른 Data access, system의 확장성이 용이하다. 예로는 DDTS, CARAT 등의 Prototype이 있다. 전반적으로는 앞서 말한 분산시스템의 특성 즉, 전체 real state를 알 수 있는 순간이 없다는 문제점, 그리고 각 Component들의 resource object 들이 분산되어 있다는 것이 일으켜 주는 문제점이 많다.

즉 Process synchronization and scheduling, Naming, Error Control, Protection, Dead lock, Resolution, Resource allocation, decentralized control 등 풀어야 할 문제점이 많으며, 많은 연구가 기대되고 있는 분야이다.

끝으로 분산처리하는 원칙을 간단히 소개하면 첫째 Data가 있는 곳에서 Processing이 이루어지게 하며 둘째, 여러곳에서 어떤 Processing이 일어나게 할 수도 있게 하여 Availability를 높이며, 세째 각 컴퓨터 시스템에 특수한 task를 주어서 전체 system에서의 설계의 단순화를 꾀한다고 말할 수 있겠다.

Ⅵ. 결 론

앞으로 국내에도 Computer 또는 Computer based machine이 많이 보급됨에 따른 이유와, 현재 소형에서 중형까지 이르는 국내 Computer 산업의 활성화를 대형컴퓨터의 의존도를 낮춘다는 이유, 정보산업화 사회에로의 필연적인 변환이라는 이유 그리고 차세대 컴퓨터 개발을 위한 기술이 밑받침이 된다는 이유 등으로 분산시스템에 대한 연구 및 개발이 매우 절실한 현실이다.

분산 시스템은 앞서 열거한 바와 같은 고유의 장점은 많으나 해결해야 할 문제점 또한 많기 때문에 복잡해진다는 특성이 있게 된다. 다양한 분산시스템 형태별로 체계적인 연구가 밑받침되어 필요한 기술개발이 이루어져야 하겠다.

참고문헌

[1] David R. Cheriton, "The V Kernel : A Software Base for Distributed Systems", IEEE Software, pp. 19-42, April 1984

[2] Richard F. Rashid, George G. Robertson, "Accent : A Communication Oriented Network Operating System Kernel:, Proc. of the 8 th ACM Symp. on Operating System Principles, pp. 64-75, April 1981

[3] Lawrence A. Rowe, Kenneth P. Birman, "A Local Network Based on the UNIX Operating System", IEEE Trans. on Software Engineering, Vol. SE-8, No. 2, pp. 137-145, March 1982

[4] Bruce Walker, et al., "The LOCUS Distributed Operating System", Proc. of the 9th ACM Symp. on Operating System Principles, pp. 49-70, October 1983

[5] Peter J. Denning, Robert L. Brown, "Should Distributed system be hidden?", IEEE, 1983

[6] L. Svobodova, et al., "Workshop on Operating Systems in Computer Networks", Operating System Review, April 1984

[7] Amnon Barak, Ami Litman, "MOS : A multicomputer Distributed Operating System", Software-Practice and Experience, Vol. 15(8), pp. 725-737, August 1985

[8] J. S. Banino, J. C. Fabre, "Distributed Coupled Actors : CHORUS Proposal for Reliability", 3 rd International Conf. on Distributed Computing Systems, October 1982

[9] Marc Guillemont, "Etude comparative de quelques systems repartis", T. S. I., Vol. 3, No. 1 pp 5-21, 1984