

지능형 컴퓨터 개발 기술

朴 治 恒

韓國電子通信研究所 컴퓨터研究部

I. 서 론

지능형 컴퓨터 개발사업이 국내에서 본격적으로 추진되자 학계, 산업계는 물론 정부부처 및 언론계에서까지 관심이 고조되고 있다. '지능형'이라는 단어가 풍기는 매력(?) 때문인 듯 싶다. 컴퓨터라는 것이 일종의 기계이긴 하지만 다른 기계와 구별되는 것은 일반기계와는 달리 인간이 장시간 골뎠하게 생각하고 계산해야 할 일을 빠른 시간내에 대신해 주는 역할을 해낼 수 있기 때문이다. 즉, 지능적인 기계인 것이다. 이런 지능적 기계인 컴퓨터에 다시금 지능형이란 단어가 붙은 컴퓨터는 과연 어떤 컴퓨터 일까.

일반적으로 지능형 컴퓨터라고 하면 사람에 따라, 응용분야에 따라 여러가지로 다르게 정의할 수 있을 것이다. 본고에서는 현재 한국전자통신연구소에서 개발중인 지능형 컴퓨터에 대해 설명하고자 한다. 지능형 컴퓨터를 한마디로 표현하면 인간과 유사한 컴퓨터라고 할 수 있다. 다시말하면 인간과 같이 보고, 듣고, 말하고, 생각하고, 판단하며 때로는 새로운 지식을 습득할 수 있는 컴퓨터인 것이다. 우리의 궁극적인 목표가 여기이긴 하지만 현재의 기술수준, 향후 기술발전 추이, 실현가능성, 시장성 등을 감안하여 현실적으로 설정한 지능형 컴퓨터의 주요 기능은 사용자 편의성, 멀티미디어 정보처리 그리고 지식정보처리의 세가지로 요약될 수 있다.

사용자 편의성이란 인간과 컴퓨터가 자연스럽게 대화할 수 있는 환경을 말한다. 예를 들면 컴퓨터 사용자가 필요한 정보를 입력시키고저 할 때 편의에 따라 키보드에 글자를 쳐 넣기도 하고 말로 하는것이 편하면 음성으로, 때로는 스크린에 나타난 메뉴를 손으로 지적하여 입력시키거나 필요에 따라 글자를 직접 써 넣거나 그림

을 그리는 등 다양한 방법을 사용할 수 있도록 하는 것이다.

멀티미디어 정보처리란 다양한 입력장치를 통해 들어온 정보 즉, 텍스트, 음성, 화상, 영상 등 멀티미디어 정보를 서로 관계된 것들은 연결시켜서 필요에 따라 디스크에 저장하거나 저장된 멀티미디어 정보를 새롭게 편집하여 출력시키거나, 텍스트 형태로 저장된 정보를 음성으로 변환시키는 미디어 변환 따위의 정보처리 형태를 뜻한다.

지식정보처리는 입력된 정보를 이해하여 기 저장된 이미지와 연결시켜 주거나 새로운 지식을 추론하거나 학습을 통하여 새로운 지식을 습득하는 등의 정보처리 형태를 의미한다.

지금까지의 설명을 요약하면 지능형 컴퓨터란 고급정보처리 컴퓨터 또는 지식정보처리 컴퓨터라고 할 수 있으며, 지식정보를 처리할 수 있기 위하여는 일차적으로 그림, 언어, 텍스트 등 비정형화된 멀티미디어 정보를 입출력할 수 있으며, 또한 멀티미디어 정보를 실시간으로 처리할 수 있는 능력을 가지고 있어야 한다.

특히, 유의할 사항은 지능형 컴퓨터란 사용자 입장에서 요구기능을 위주로 정의된 컴퓨터이며, 기술적인 측면에서 정의된 컴퓨터가 아니라는 점이다. 그런 점에서 지능형 컴퓨터와 인공지능 컴퓨터를 혼동해서는 안된다. 지능형 컴퓨터 구현을 위해서 필요한 기술의 하나가 인공지능 기술이며, 그 밖에도 여러가지 기술 즉, 병렬처리기술, 신경망 기술, 데이터 압축 및 복원기술 등 다양한 기술이 요구되는 것이다.

지능형 컴퓨터는 크게 5가지 즉, 기본 하드웨어, 기본 소프트웨어, 멀티미디어/지식정보처리 유틸리티, 멀티미디어 사용자 인터페이스, 고속 통신망 인터페이스로

구성되어 있다. 본고에서는 이중 특히 기본 하드웨어 및 기본 소프트웨어를 중점적으로 기술 하고자 한다.

II. 지능형 컴퓨터 구조

1. 하드웨어 구조

지능형 컴퓨터의 하드웨어는 대규모 멀티미디어 데이터 및 지식 정보를 고속으로 처리할 수 있는 platform으로서 이에 적합한 대규모 병렬처리 구조를 갖는다. 지능형 컴퓨터 하드웨어는 그림 8과 같이 실제적인 프로세싱을 수행하게 되는 computation unit, 많은 양의 정보를 저장할 수 있는 mass storage unit, 사용자 인터페이스와 고속 통신망을 통해 외부와 연결하기 위한 multimedia I/O unit, 전반적인 시스템의 관리 및 통제를 하는 system control unit, 그리고 이들 unit들간의 연결을 위한 inter-unit connection 등으로 구성되어 있다.

Computation unit은 지능형 컴퓨터가 수행하는 업무의 궁극적인 computation을 빠르게 하기 위한 unit이다. 지능형 컴퓨터는 고압성보처리를 주 업무로 하며, 이는 구체적으로 multimedia data processing 및 knowledge processing을 효과적으로 수행할 수 있는 시스템을 의미한다. 따라서 지능형 기본 하드웨어의 computation unit은 multimedia data 및 knowledge processing을 효과적으로 지원할 수 있는 구조를 가져야 하며, 아울러 기존 시스템에서 제공하는 일반적인 processing도 효과적으로 처리할 수 있는 구조를 가져야 한다.

이러한 상위 레벨의 processing을 수행하기 위하여 요구되는 세분화된 operation들로는 pattern matching, geometrical transformation, data compression/decompression, inference 및 process synchronization등이 포함된다. 이러한 operation을 수행하기 위하여 하위 레벨에서 요구되는 처리 기능은 symbolic data, floating-point/integer data에 대한 기본적인 연산이며 computation unit의 구조는 이를 고속으로 지원하여야 한다. 이외에도 지능형 컴퓨터에서는 object-oriented paradigm의 비중이 클 것이라 예상이 되므로 기본 하드웨어에서 이를 지원하기 위한 구조와 primitive hardware mechanism을 제공하여야 할 것이다.

Computation unit은 이들 processing을 고속으로 제공하기 위하여 다수의 프로세서가 존재하는 massively parallel processing 구조를 갖고 있다. Processing node는 symbolic processor, multimedia processor, inter-processor communication controller, memory 및 memory

controller등을 포함하며 symbolic processor는 지식정보 처리와 일반적인 처리를 담당하고, multimedia processor는 data compression등 멀티미디어 데이터 처리를 담당한다. Inter-processor communication controller는 processing node 간의 통신을 담당하며 효율적인 parallel processing이 일어나도록 지원한다.

Processing node 간의 연결 구조는 멀티미디어 지식정보의 병렬 처리를 효율적으로 지원하면서 통신 채널의 bandwidth가 높고 latency가 작은 구조를 사용할 예정이다. 앞으로 체계적인 성능 분석을 통하여 최종 구조가 확정될 것이다. Routing 기법으로는 Ametek 2010 컴퓨터 등에서 사용되어 그 성능이 입증된 wormhole routing이 고려되고 있다.

Mass storage unit은 storage space를 제공하는 disk와 이를 관리하고 다른 unit들과의 효과적인 데이터 전송을 수행하기 위한 disk controller등으로 구성된다. I/O unit는 입력된 multimedia 데이터에 대한 preprocessing/post-processing을 제공하기 위한 multimedia processor와 여러가지 입력 device들과의 interface를 제공하는 device interface 및 I/O controller등으로 구성된다.

한편 지능형 컴퓨터의 networking 기능을 제공하기 위한 고속 통신망 interface unit은 LAN interface와 데이터 송수신에 필요한 buffer memory, 그리고 전체 network을 관리하고 데이터 송수신을 제어하는 protocol processor 등이 포함된다. Inter-unit connection은 실시간 비디오 전송과 같은 수십 Mbyte/sec의 높은 bandwidth의 데이터 전송과 low-latency를 갖고 각 unit간의 통신을 담당한다.

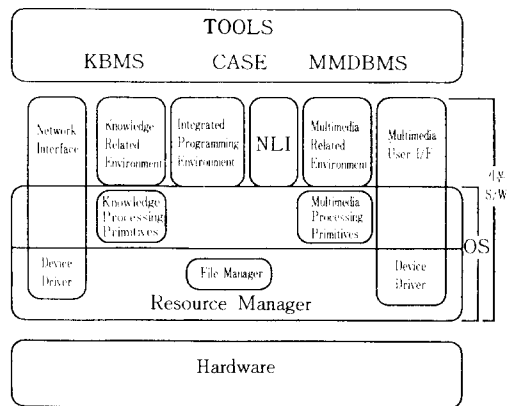


그림 1. 지능형 컴퓨터 기본 S/W의 구조

2. 소프트웨어 구조

지능형 컴퓨터의 기본 시스템은 하드웨어를 비롯하여 운영체제, 프로그래밍 환경, 사용자 인터페이스, 네트워크 인터페이스 그리고 멀티미디어 및 지식정보 처리를 위한 기본적인 유틸리티들을 포함한다.

기본 소프트웨어는 기본 시스템의 구성 요소들 중에서 그림 1과 같이 하드웨어를 제외한 나머지 부분으로 구성 된다.

기본 소프트웨어는 지능형 컴퓨터 사용자에게 지식정보 처리 및 멀티미디어 처리 업무를 위한 가장 기초적인 환경을 제공하며, 지능형 컴퓨터의 자원을 관리하는 역할을 한다.

기본 소프트웨어에서 제공하는 통합된 프로그래밍 환경은 사용하기 편리한 프로그래밍 환경을 제공하는 역할을 한다. 지능형 컴퓨터의 통합된 프로그래밍 환경은 그림 2에서와 같이 editor, translator, debugger, library 및 이를 관리하는 library manager 등으로 구성되며, 이들 도구들은 프로그래머가 프로그래밍 하기 편리하도록 object의 개념으로 통합되어 구성된다.

지능형 컴퓨터에서는 또한 서로 다른 프로그래밍 언어들을 혼합하여 시스템을 구성할 수 있는 기능을 제공하기 때문에 각각의 언어를 위한 요소들(translator, debugger, library 등)을 통합 관리하는 language interface를 제공한다. 따라서 지능형 컴퓨터의 사용자는 ICOP(L(개발하고자 하는 프로그래밍 언어의 가칭)을 중심으로 function(Lisp), logic(Prolog), procedural(C, Pascal, ...)등과 같은 여러가지 paradigm의 언어들을 필요한 부분에 혼합하여 사용할 수 있다.

Library manager는 프로그래머가 사용할 수 있는 lib-

rary operation들의 모임인 object library를 관리하는 역할을 한다. Library manager는 각각의 언어마다 구현되어 있는 translator, debugger등과도 유기적으로 연관되어 있다.

지능형 컴퓨터의 운영체제는 이와같은 관리의 역할 이외도 데이터를 처리하는 기능을 포함하고 있다. 즉, 사용자의 고급 응용업무 처리를 위한 프리미티브 오퍼레이션을 운영체제의 차원에서 제공한다. 이에따라, 지능형 컴퓨터의 운영체제는 그림 3에 나타난 바와 같이 대략적으로 application specific part, resource manager, hardware interface part 그리고 guest OS module의 네가지로 구성되어 있다.

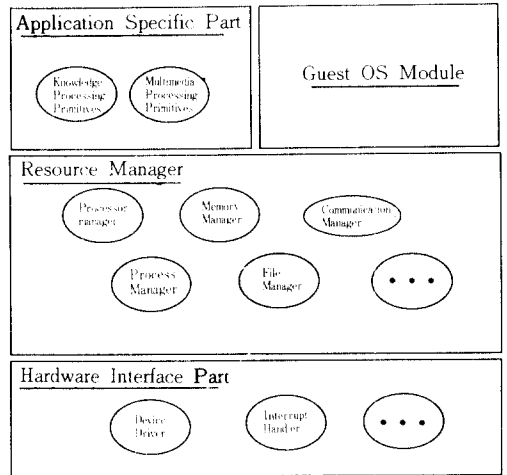


그림 3. 지능형 컴퓨터 운영체제의 개념 모델

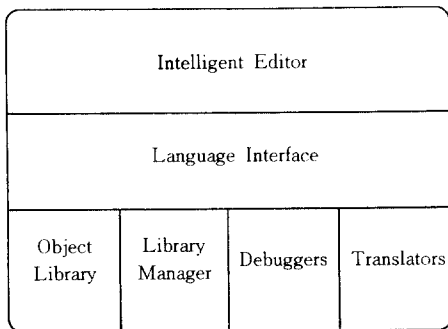


그림 2. 통합된 프로그래밍 환경의 개념 모델

3. 표준화 및 응용 소프트웨어 지원

기본 소프트웨어를 개발하는데 있어서 고려되어야 하는 사항, 특히 설계에 영향을 주는 사항은 기존의 시스템에서 수행되는 소프트웨어 package와의 호환성, 표준화의 수용 또는 표준화 안의 제시 등이다.

먼저 compatibility는 source level의 compatibility와 binary level의 compatibility를 고려할 수 있다. 전자의 경우는 기존의 소프트웨어가 수행되는 환경을 운영체제의 kernel에서 대부분 제공해 주어야 한다. 따라서 지능형 컴퓨터 운영체제의 설계원칙으로 고려하고 있는 kernel의 최소화 및 운영체제의 모듈화에 부합되지 않는다고 할 수 있다.

후자의 경우는 kernel의 설계에 거의 영향을 미치지 않으며, 기존의 소프트웨어가 수행되는 환경을 kernel 위에 하나의 모듈로 구현함으로써 제공할 수 있다. 즉, emulation library 등의 제공으로 compatibility를 유지할 수 있다. 이러한 접근과 운영체제의 기본적인 설계원칙을 고려하여 기본 소프트웨어에서 제공하는 시스템 소프트웨어는 binary compatibility를 제공하도록 한다.

기본 소프트웨어에서 프로그래밍 언어와 운영체제는 표준화와 밀접한 관련이 있다. 프로그래밍 언어에서는 여러 paradigm의 언어를 지원하기 위한 language interface를 설계하여야 하는데 현재 각 paradigm에서 표준화된 프로그래밍 언어로 제안되고 있는 언어를 수용하여야 한다. 또한 이들 언어에서 제공하고 있는 library를 수용하여 library manager에서 관리하여야 한다.

운영체제의 표준화는 명령어의 표준화, machine independent한 부분의 표준화, distributed computing environment의 표준화 등과 같이 표준화안이 이미 제시되어 상당부분 구현되어 상용화 되고 있다. 따라서 이미 표준화되어 사용되는 부분에 대한 지원을 하여야 하며, 이는 compatibility를 제공하는 것과 관련하여 고려되어야 한다.

표준화를 위한 활동은 현재까지 우리나라에서 이루어졌던 것처럼 단지 국제 표준화의 동향을 파악하고 그 표준안을 수용하는데 그치지 말고 국제 표준화를 이끌고 나가는 주체가 되도록 하여야 할 것이다. 국제 표준화를 위한 전략으로는 다음 그림 4에서와 같이 지능형 운영체제 위에 UNIX의 명령어 및 라이브러리들을 제공하여 기존에 사용되고 있는 대부분의 응용 프로그램 및 유틸리티 프로그램들을 사용할 수 있게 하여야 할 것이다. 그리

고 UNIX를 제외한 부분에서도 점차 국제 표준화가 될 수 있도록 국제 표준화 활동에 적극 참여하여야 할 것이다.

Ⅲ. 요소 기술

1. 하드웨어 기술

지능형 컴퓨터 하드웨어 개발에 필요한 기술은 병렬 구조 설계 기술, 하드웨어 서브시스템 설계 기술, VLSI 프로세서 설계 기술, 구현 기술 등으로 크게 구별된다.

병렬 구조 설계 기술은 멀티미디어 지식정보 처리를 효율적으로 지원하기 위한 하드웨어의 주요 구조를 설계하는 것으로 computation unit 내에서 processing node의 구조와 inter-processor communication topology를 설계하는 것과 inter-unit connection을 설계하는 것이다. 각각에 다양한 방식이 사용될 수 있기 때문에 가장 효율적으로 멀티미디어 지식정보 처리를 지원하는 구조를 결정하기 위한 성능 분석이 중요한 요소이다.

하드웨어 서브시스템 설계 기술은 mass storage controller, multimedia I/O unit, network interface unit 등을 설계하는 것이다. 멀티미디어 데이터를 disk와 같은 mass storage에 실시간으로 저장하기 위해서는 disk caching 또는 disk array를 이용한 parallel access 등의 기법이 필요하며 이를 지원하기 위한 mass storage control 기술이 중요하다. Multimedia I/O unit은 사용자와 교신하기 위한 데이터의 입출력 및 preprocessing/postprocessing을 담당하는 것으로서 멀티미디어 데이터의 실시간 처리와 synchronization 등을 지원하기 위한 I/O unit 하드웨어 구조 설계가 중요한 기술이다. 또한 network interface unit도 400Mbps 이상의 전송속도를 갖는 고속 통신망과 지능형 컴퓨터를 연결시키는 것으로 통신 protocol의 고속 처리와 멀티미디어 데이터 스트림의 synchronization을 효율적으로 지원하기 위한 하드웨어 구조 설계가 중요한 문제이다.

VLSI 프로세서 설계 기술은 멀티미디어 지식정보 처리, multimedia I/O, multimedia 통신, processing node 간의 통신에 필요한 VLSI 프로세서 중에서 상용 프로세서가 존재하지 않거나 미흡할 경우 이를 설계하는 기술로서 지능형 컴퓨터에서 설계할 필요가 있는 프로세서로는 multimedia processor, 지식정보 처리를 위한 symbolic processor, inter-processor communication controller, multimedia communication protocol accelerator, 고속 LAN adapter 등이 있다. 이러한 프로세서를 설계하

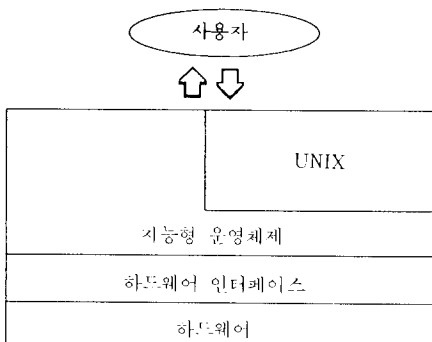


그림 4. 표준화를 위한 지능형 운영체제 구조

는 데는 프로세서의 functional specification을 최소의 회로를 사용하여 최고의 처리 속도를 구현하기 위한 VLSI microarchitecture의 최적 설계 기술이 필수적이다.

지능형 컴퓨터 하드웨어에서 중요한 구현 기술은 massively parallel 구조를 compact하게 구현하기 위한 것으로서 이에 MCM(multichip module)packaging 기술과 optical interconnection 기술이 있다. MCM은 processing node module 이나 inter-processor communication adapter module 등에 사용될 수 있다. Optical interconnection은 optical fiber를 사용하여 processing node 간의 연결이나 inter-unit connection을 구현하는데 사용될 수 있으며 optical communication이 갖는 높은 bandwidth 때문에 적은 수의 connection으로 높은 전송 속도를 얻을 수 있는 장점이 있다.

2. 소프트웨어 기술

기본 소프트웨어를 위한 개발 기술은 운영체제 개발을 위한 기술들과 프로그래밍 환경개발을 위한 기술로 나눌 수 있다. 이중 운영체제를 위한 기술들에 대해 살펴보면, 한글처리를 위한 기술, 각종 자원의 관리 기술, 다른 시스템과의 통신 기술, 그리고 멀티미디어 입출력을 위한 디바이스 관리 기술 등으로 나누어 볼 수 있다. 한글처리를 위한 기술로는 한글 코드를 비롯하여 문자, 음성, font등에 이르기까지 다양한 한글 처리 기술이 필요하며, 각종 자원의 관리 기술로는 대용량 멀티미디어 화일 관리 기술, 지식 정보처리기술, 실시간 처리기술들이 있다. 또한 다른 시스템과의 통신기술에 있어서는 대용량 데이터의 근거리/원거리 고속 전송, 멀티미디어 데이터의 전송기술등을 들 수 있으며 디바이스 관리 기술로는 지능형 컴퓨터를 위한 scanner, speaker, mic, audio, VTR 시스템등의 주변기기를 관리하는 기술들을 들 수 있다. 또한 분산처리 및 병렬처리 기술들이 제공되어야 하며 이들을 위한 load balancing 기술 inter process communication 기술들이 필요하다.

프로그래밍 환경을 위한 기술들은 프로그래밍 언어 설계기술, visual language들을 위한 그래픽 기술, 그리고 프로그래밍 언어를 구현하기 위한 컴파일러/translator 개발 기술들이 필요하다. 프로그래밍 언어를 설계하기 위해서는 멀티미디어 및 지식 정보처리를 위한 언어의 paradigm의 정의, 프로그래밍 언어에서 사용할 수 있는 데이터의 형태 및 이들을 위한 각종 operation들을 설계하여야 할 필요가 있다. 또한 이들 프로그래밍 언어 및 디버거들은 병렬처리 환경에서 제공되기 때문에 병렬처리

기술도 제공되어야 한다. 그리고 프로그램의 재사용성을 높이기 위한 standard library 및 user library 관리 그리고 이들을 위한 language 개발 기술들이 필요하며 library 및 프로그래밍 언어에서 제공하는 각종 object를 관리하는 기술이 필요하다 할 수 있다.

IV. 개발방법

1. 개발체계

여러가지 기술이 복합되어 사용되고 많은 모듈이 복잡하게 연결되어 이루어지는 대형컴퓨터 시스템 개발은 목적을 달성하기 위한 방법 및 절차 즉, 개발체계가 무엇보다도 중요하다 하겠다. 국내에서도 TDX 및 행정전산망 주전산기 개발을 통하여 개발체계의 중요성을 인식하게 되었다. 두 대형 프로젝트에서 사용된 개발체제는 water fall 모델로써, 개발코저 하는 시스템의 reference machine이 있을 경우 효과적인 개발체계라 할 수 있다.

지능형 컴퓨터는 구체적인 reference machine이 없기 때문에 단순한 water fall 모델을 따르지 않고 rapid prototyping 모델을 혼합한 새로운 개발체계를 사용하고 있다. 그림 5는 지능형 컴퓨터 개발체계를 그림으로 나타낸 것이다. 이 개발체제는 기본적으로 water fall 모델에

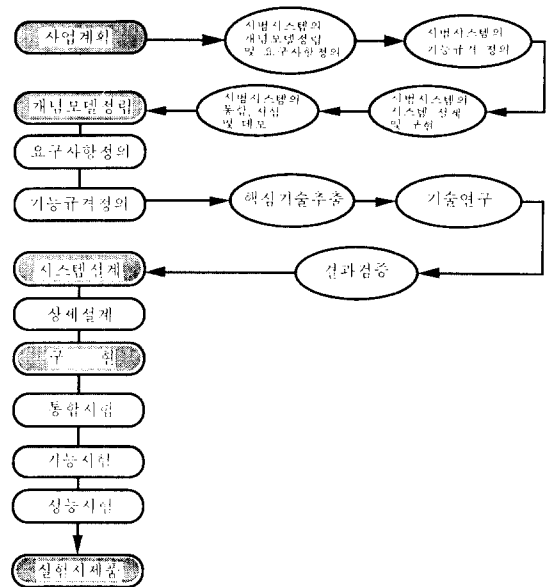


그림 5. 지능형 컴퓨터 개발 순기 모델

근거한 개발 순기과정을 지치지만 내부적으로는 development, research 그리고 다시금 development 과정을 반복함으로써 3번의 cycle을 도는 형태인 D-R-D 모델인 것이다.

첫번째 cycle에서는 rapid prototyping에 의한 시범시스템의 설계와 구현을 통하여 정확히 정의되지 않은 target machine의 개념모델을 정립하는 것을 목적으로 하고, 두번째 cycle에서는 정립된 개념모델을 분석하여 target machine의 요구사항과 기능규격을 정의하고, 또한 각 분야별 핵심기술을 추출한 다음 요구기술을 해결하기 위한 연구업무를 수행하고 이의 결과를 바탕으로 시스템을 설계하는 것을 목적으로 한다. 끝으로 세번째 cycle에서는 설계된 시스템을 각 모듈별로 상세설계를 수행하고 이를 구현하여 전체 시스템으로 통합하고 기능 및 성능시험을 거쳐 실험시제품을 완성하는 것을 목적으로 하였다.

2. 하드웨어 개발방법

지능형 컴퓨터 하드웨어 개발은 rapid prototype의 설계와 구현을 통하여 target machine의 기본개념 및 요구사항등을 분석하며 본격적인 하드웨어 개발의 reference machine 및 test bed로 이용한다. 지능형 컴퓨터의 본격적인 하드웨어 개발은 구조의 복잡성을 고려하여 각 유니트 단위로 단계별로 이루어질 것이다. 개발의 순서는 각 유니트의 중요성과 난이도를 염두에 두고 결정된다. 지능형 컴퓨터의 핵심이 되는 computation unit가 먼저 구현이 되고, 다음이 mass storage unit, multimedia I/O unit, 그리고 control unit가 개발될 예정이다. 또한 각 유니트간의 연결(inter-unit connection)은 다량의 정보가 고속으로 이동할 수 있는 통로가 제공되어야 하므로, 각 유니트가 설계구현될 때 유니트간의 연결방식 또는 연결 네트워크의 결정은 매우 중요한 요소이다.

1) Rapid prototype 시스템의 개발

Rapid prototype의 개략적인 모습은 그림 6과 같으며 모두 4개의 unit들로 구성된다. 이 unit들은 크게 main system과 remote system의 두가지 그룹으로 나뉘어 질 수 있는데 main system은 computation unit, mass storage unit, multimedia I/O unit으로 구성되어 지능형 컴퓨터의 역할을 수행하고, remote unit만으로 구성되는 remote system은 main system에 연결되어 service를 받는 remote user system의 역할을 수행하게 된다. 이 두개의 그룹으로 이루어지는 시범시스템은 각기 고유한 기능을 갖는 서로 독립적인 4개의 unit들이 high speed network

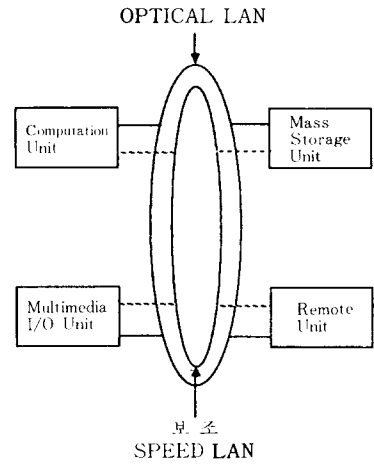


그림 6. Rapid prototype 시스템

으로 연결되어 구성되며 전체적으로 분산 시스템의 모양을 갖게 된다.

2) Computation unit의 개발

이 단계에서 개발할 시스템의 모습은 그림 7과 같다. Computation unit는 다수의 프로세서와 프로세서들간의 통신로로 구성되어 있다. 기본적으로 이 단계에서 결정 또는 개발되어야 할 것은 프로세서의 성능과 종류, 프로세서간의 연결 토폴로지, 메시지의 라우팅(routing) 방식등이다. Computation unit의 기능을 점검하기 위해서 disk I/O나 user I/O는 host machine을 통해서 하게 되며, control unit의 기능도 host machine에서 하게 된다.

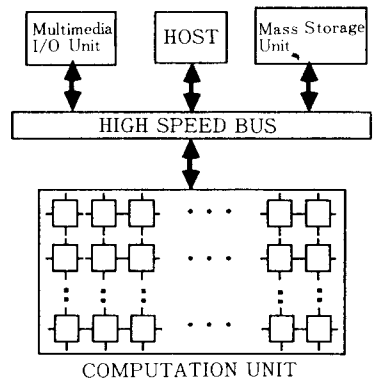


그림 7. 중간 단계 하드웨어 구조

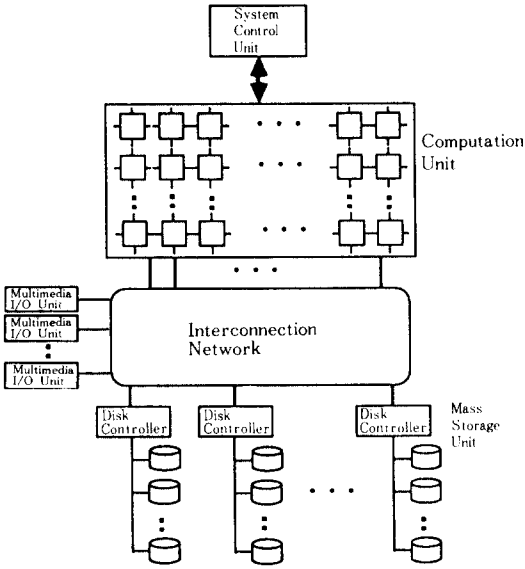


그림 8. 최종단계 하드웨어 구조

3) 기타 하드웨어 unit 개발

이 단계에서는 그림 8에 나타나 있듯이 mass storage unit, multimedia I/O unit, inter-unit connection, system control unit 등 전체 지능형 컴퓨터 하드웨어를 구현한다.

대량의 멀티미디어 정보 및 지식정보를 보관하기 위해, 그리고 시스템이나 사용자의 요구대로 실시간으로 데이터를 입출력 시키기 위해서는 conventional disk들은 적당하지 않다. 아직 새로운 개념의 mass storage device는 개발되지 않고 있으나 적은 용량의 값싼 hard disk의 array를 이용해서 reliability와 performance가 높은 storage device를 구성하려는 연구가 활발하다.^{[1][2][3][4]} 이 외에도 conventional disk를 이용하되 cache 이론을 응용한 disk caching^{[5][6]}, write time을 줄이기 위해 고안된 log-structured file system^[7]등의 연구도 상당한 주목을 받고 있다.

이런 연구들은 근본적으로 processor의 처리속도의 증가 및 memory의 고집적화와 고속화에 비해 속도나 용량의 발전이 느린 mass storage device가 전체 computer system의 bottleneck이 되고 있어 이를 타개하기 위한 것이지만 지능형 컴퓨터의 mass storage를 위해서도 유용하게 이용될 수 있을 것으로 생각된다. 따라서 이 분야에 대한 집중적인 연구가 수행되어 멀티미디어 및 지식

정보의 효율적인 보관, 전송이 가능하게 되어야 할 것이다. Mass storage unit와 computation unit 사이의 연결 configuration의 요구사항은 다음과 같다.

- Computation unit의 disk I/O 담당 node는 모든 disk를 access할 수 있어야 한다.
- 필요에 따라 mass storage unit이 disk의 수를 증가(혹은 감소)시킬 수 있어야 한다(scalability).
- Mass storage unit과 computation unit 사이의 데이터 전송시에 disk interconnection network에 병목현상이 발생해서는 안된다.

이 단계에서는 disk unit이 선택 또는 개발될 것이며, 위의 요구 사항을 충족시키는 interunit connection network이 구현된다.

Multimedia I/O unit의 구현을 결정하기 위해 먼저 해야 할 일은 user I/O의 traffic을 분석하는 것이다. User I/O의 traffic은 다음과 같이 2가지로 나누어 생각해 볼 수 있다.

- Command traffic : user가 시스템에 보내는 명령들로 비교적 짧은 character string(keyboard command, pointing device등) 이나 speech data(speech command의 경우)들로 구성된다.
- Multimedia, 지식정보 data traffic : user가 시스템에게 혹은 시스템이 user에게 보내는 모든 종류의 data들로 textual data의 경우와 같이 비교적 소량일 경우도 있지만 일반적으로 부피가 큰 정보들이다.

Multimedia I/O unit와의 연결을 위한 요구사항은 mass storage unit과의 연결을 위한 그것과 비교할 때 후자의 경우는 내용량의 데이터 입출력과 빈번한 disk access로 특성지워지는 반면에 전자의 경우는 빈번한 소용량의 command/data의 입출력, 비교적 간헐적인 내용량의 실시간 출력 및(비실시간) 입력 등으로 특성지워진다. 따라서 mass storage unit interface를 위한 요구사항보다는 덜 제한적이지만 실시간 처리를 위해 독립적인 하드웨어로 구성될 것이다.

현재 제안되고 있는 대부분의 병렬처리 시스템들의 경우 parallel processor 들은 back-end로 이용되고 이것의 control(task scheduling, task assignment, resource management, user 및 peripheral device control 등)은 하나의 host machine에서 처리한다. 이런 configuration은 병렬성이 높은 반복 계산을 위주로 하는 과학계산용 컴퓨터에는 적합할 지 모르나 지능형 컴퓨터와 같이 다수의 사용자, multimedia data의 입출력 및 보관, 다양한 정보의 처리 등과 같은 특성을 갖는 시스템에서는 host와 back-

end 사이의 communication link가 병목이 되기 쉬울 뿐 아니라 host machine에서 처리, 결정해야 할 task들이 너무 많아 back-end를 효율적으로 활용하기 힘들 것으로 보인다.

다시 말하면, multimedia 처리는 on-line 또는 실시간 처리가 요구되는데, 모든 전처리(pre-processing)를 host machine에서 담당하고 back-end machine이 모든 task의 off-line 방식으로 처리하게 하는 것은 바람직한 선택이 아니라고 보여진다. 반면에 host, back-end의 구분을 없애고 computation unit 자체가 순수한 stand alone system이 되게 구성한다면 분산 제어(distributed control) 시스템의 모습을 갖게 되는 데 이 경우에는 O/S를 구현하기 힘들 뿐 아니라, 구현한다 하더라도 processing node들이 처리해야 하는 control overhead, communication overhead가 많아 비효율적일 것이다.

따라서 지능형 컴퓨터의 하드웨어 구조는 이 두 가지의 극단적인 구조를 절충해서, 중앙 집중적으로 전 시스템을 제어관리하는 system control unit의 개념을 도입 구현하게 될 것이다.

3. 소프트웨어 개발방법

기본 소프트웨어중 운영체제의 개발은 먼저 기존의 UNIX 운영체제 위에서 수행될 수 있는 운영체제를 위한 프로그램을 개발한 후 차례로 UNIX 운영체제의 기능을 대체해 나가는 방향으로 추진될 것이다. 운영체제의 여러가지 기능 중 가장 급급히 해결하여야 하는 것은 멀티미디어 처리를 위한 화일 시스템 개발이다. 이와 동시에 멀티미디어 입출력을 위한 각종 장비들을 UNIX에 붙이는 일을 하여야 한다. 초기의 멀티미디어 화일 시스템은 UNIX 위에서 수행될 것이며 점차적으로 텍스트 처리 위주의 UNIX 화일 시스템을 대체해 나갈 것이다. 물론 이를 위해서는 대용량 화일들을 저장할 수 있는 저장장치가 UNIX 상에서 사용 가능하여야 할 것이다. 멀티미디어 화일 시스템의 개발후에 새로이 개발된 메모리 관리 시스템, 프로세서 관리 시스템 등이 UNIX 시스템의 해당 부분을 대체하게 된다. UNIX의 기본적인 대부분의 기능이 포함된 새로운 운영체제가 설계되면 real-time 기능 및 병렬처리를 위한 interprocess communication 기능이 추가될 것이다. 이러한 새로운 운영체제가 개발된 후에 UNIX와의 인터페이스를 위한 프로그램들이 이 운영체제상에서 구현될 것이다.

프로그래밍 언어개발에 있어서 선행되어야 할 일은 현재 기존의 시스템에서 잘 수행되고 있는 c++ 언어 위에 library 형태로 멀티미디어 데이터를 위한 type 및

operation들을 정의하여 프로그래머가 이 언어에서 제공하는 기능을 사용하게 하는 것이다. 그 이유는 구현이 용이할 뿐만 아니라 새로운 데이터 type 및 operation을 이용한 프로그래밍이 가능하고 새로운 프로그래밍 언어의 형태를 보다 쉽게 파악할 수 있기 때문이다. 그런 연후에 이 새로운 syntax 및 semantics에 적합한 interpreter가 개발될 것이다. 이러한 방법은 새로운 기능의 삭제 및 첨가가 용이하며 기존 시스템에서 수행시켜볼 수 있는 잇점이 있다. 최종적으로 새로이 정의된 CPU instruction을 generate할 수 있는 compiler를 개발하는 것이다.

V. 결 론

지능형 컴퓨터란 특수한 목적을 위해 필요한 컴퓨터가 아니다. 컴퓨터가 일반화되고 사용 용도가 다양해지면서 자연스럽게 요구되는 멀티미디어화와 지능화를 수용하는 컴퓨터이다. 가정에서는 오디오, 비디오, TV 및 컴퓨터가 통합되어 가고 있고 지능망, ISDN 등 통신망이 지능화 고속화 되어가고 있는 상황 속에서 컴퓨터가 발전해 나가야 할 방향인 것이다.


특히, 이 시점에서 지능형 컴퓨터 개발이 강조되는 것은 이 분야의 시장이 엄청나게 클 뿐만 아니라 선진국에서도 본격 개발을 이제 막 시작한 단계이기 때문에 컴퓨터 기술이 낙후되어 있는 우리도 이러한 전환기적 호기를 놓치지 않고 적극적으로 연구개발에 참여함으로써 기술선진국 진입을 앞당길 수 있을 것이기 때문이다.

參 考 文 獻

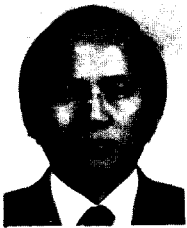
- [1] D. A. Patterson et al., "A Case for Redundant Arrays of Inexpensive Disks(RAID)", *ACM SIGMOD Conference*, pp. 109-116, June 1988.
- [2] A. L. N. Reddy, "Performance Evaluation of Multiple-Disk I/O Systems", *1989 International Conference on Parallel Processing*, vol. 1. pp. 315-318, 1989.
- [3] W. E. MEADOR, "Disk Array Systems", *1989 IEEE Spring COMPCON*, pp. 143-146, 1989.
- [4] Spencer Ng, "Some Design Issues of Disk Arrays", *1989 IEEE Spring COMPCON*, pp.137-142, 1989.
- [5] P. J. Jalic et al., "Caching and Other Disk Access Avoidance Techniques on Personal Computers", *Communications of the ACM*, pp. 246-255, Feb. 1989.

[6] J. Menon et al., "The IBM 3990 Disk Cache",
 1988 IEEE Spring COMPCON, pp. 146-151,
 1988.

[7] John Ousterhout, "Beating the I/O bottleneck : A

case for log-structured file system", *ACM : Operating Systems Review*, vol. 23, no. 1, pp. 11-28, Jan. 1989. 

筆者紹介



朴 治 恒

1947年 1月 12日生
 1974年 서울대학교 공과대학 응용물리학과 졸업(학사)
 1980年 한국과학원 전산학과 졸업(석사)
 1987年 프랑스 파리 6대학 Computer Science (공학박사)

1983年 9月~1984年 12月 프랑스 국립컴퓨터연구소(INRIA)에서
 데이터 베이스 Machine SABRE 개발 프로젝트 참가

1974年 3月~1978年 2月 한국과학기술연구소(KIST) 전산실 연구원

1978年 3月~1985年 6月 한국전자기술연구소(KIET) 시스템부 선임연구원

1985年 7月~현재 한국전자통신연구소(ETRI) 책임연구원, 현재 컴퓨터연구부 연구위원

주관심분야 : Distributed architecture, Parallel architecture, Next generation computer system,
 Multimedia information processing DBMS, OLTP