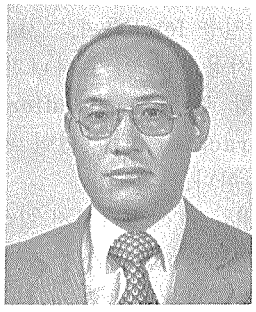


吳 吉 祿
韓國電子通信研究所
컴퓨터연구부장 / 工博

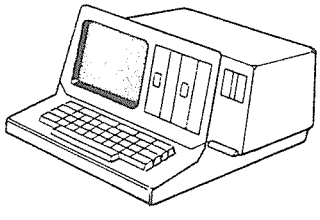
Bit-Map Graphics를 이용한 한글WPS개발

본 논문에서는 UNIX O.S하에서 사용 가능한 Bit-map graphics를 이용한 한글 워드프로세서의 구조 및 개발 방법에 대하여 설명을 한다. 이 워드프로세서는 크게 스크린 에디터 및 포매터로 구성되어 있으며 여러 종류의 한글 터미널과도 비교적 쉽게 접속이 가능하며 UNIX O.S 위에 쉽게 Porting 될 수 있게 개발되었다. 이것을 더 확대하여 한자처리 및 그래픽 처리 등도 쉽게 첨가될 수 있을 것이다.

1. 서론

UNIX는 마이크로 컴퓨터 혹은 미니컴퓨터의 O.S로서 여러 종류의 Hardware와도 쉽게 호환성을 가지기 때문에 최근 들어 전 세계적으로 사용이 증가되고 있으며, UNIX하에서 사용이 가능한 많은 Software들이 개발됨에 따라 UNIX의 사용은 컴퓨터 이용자에게 많은 잇점을 주고 있다. UNIX는 이제 학교 및 연구소에서 개발 tool로서 뿐만 아니라 다방면으로 사용이 되고 있으며 우리나라에서도 2～3년 전부터 사용이 되어 지금은 많은 관심을 끌고 있다.

본 논문에서는 이러한 환경을 가진 UNIX O.S하에서 사용 가능한 한글 워드프로세서의 개발에 대하여 이야기한다. 이 한글 워드프로세서는 크게 스크린 에디터와 포매터로서 구성되어 있으며 한글 display를 위하여 한글 터미널의 사용은 물론 Bit-map graphics방식을 이용하여 High resolution workstation에서 run되게 개발을 하며, 어떠한 UNIX machine하에서도 Source의 큰 변환없이 여러 종류의 한글 터미널과 쉽게 접속되어 사용 가능하게 개발을 한다.



개발에 앞서 여러가지 스크린 에디터 및 포매터 (Vi, Emacs, Alpha Vue, Ace, Word star, Qnix fmt 등) 들의 기능을 비교 분석하였으며,

UNIX 환경에 적합하고 최근 MIT, 벨 연구소 및 학교 등에서 많이 사용되고 있는 Emacs를 참조 모델로 하여 한글 스크린에디터를 개발하였다. 본 개발의 진행상 먼저 한글 터미널을 사용하여 수행이 될 수 있게 개발을 한 다음 한글 Bit-map display System을 개발하여 Workstation에서 수행되게 한다.

2. 한글 스크린에디터의 구조

스크린에디터는 워드프로세서의 한 부분으로 스크린상에 Cursor를 옮겨다니며, 마치 사용자가 화면상에서 서류의 일부분을 보면서 서류를 작성 혹은 수정하는 것처럼 해주는 소프트웨어이다. 스크린에디터는 크게 Command handler, Buffer Handler, Display handler 및 Input/output handler로 구성되어 있다.

Command handler는 Input/output handler로부터 한글 터미널에서 친 한 문자를 받아서 그 문자가 명령어인지 아니면 text의 일부분인지를 알아내며 만약 명령어일 경우에는 각각의 명령처리 routine으로 제어를 넘기며, text의 일부분이면 내부 Text buffer에 저장하고 Display handler를 통하여 변화된 화면을 다시 구성시킨다. 이 Command handler에서는 에디터 명령어에 대응되는 명령처리 routine의 주소 table을 가지고 있으며 각각의 명령처리 routine은 Buffer handler 및 Display handler의 각 routine을 call하여 지정된 명령을 수행시킨다. 사용자는 Command handler의 각 명령어 처리 routine의 주소 table의 내용을 변경시키는 명령을 사용하여 자기가 원하는 명령어에 각 명령을 대응시킬 수 있다.

Buffer handler는 사용자가 친 text를 내부 Text buffer에 저장하며 내부 buffer가 완전히 차 있을 때는 disk에 일시적 파일을 만들며 주기억장치 속의 buffer와 disk상의 buffer파일을 제어한다. 만약 사용자가 momory 상에 없는 text의 일부분을 요구할 때는 Buffer handler에 의해 disk에서 그 부분을 찾아서 memory 상의 buffer에 적재한 뒤 사용자에게 보내준다. 이 Buffer handler에서는 동시에 여러개의 파일을 Open하여 쓰므로 여러개의 파일을 작성 혹은 수정하기에 편리하게 되어 있다.

Display handler는 스크린에디터에서만 가지고 있는 부분으로 사용자로 하여금 화면상에 작성중인 서류의 일부를 실제로 보면서 서류를 작성하는 것처럼 해주는 routine이다. 이 Display handler의 주된 기능은 각 터미널마다 다르게 정의된 제어코드 및 Escape 연속 code를 가지고 있는 termcap 파일을 fetch하며, 사용자가 여러가지 명령어를 사용하여 옮겨 다니거나 화면의 일부분을 수정할 때 Internal buffer의 내용 수정과 함께 interactive하게 화면이 재구성되게 해주는 역할을 한다.

Input/output handler는 에디터에서 일어나는 모든 입출력에 관계되는 routine들의 종합으로서 터미널 및 disk 입출력을 관장한다. 이 부분은 한글 처리의 추가되는 부분으로 터미널 입출력의 가장 하위 Level에 한글 Automata 및 코드변환 routine이 첨가되어 있다. 한글 Automata는 한글 터미널에서 입력되는 코드를 받아서 에디터의 Text buffer에 저장시키는 2-byte 내부 코드로 변환시키며, 반대로 내부 코드를 한글 터미널용 코드로 변환시키는 작업을 한다.

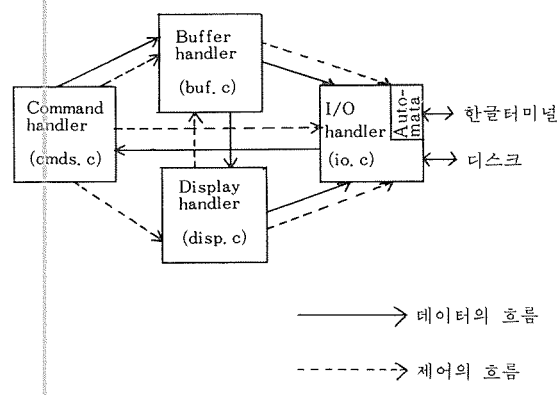


그림 1. 한글 스크린에디터의 블록 구조

위에서 설명한 구조로 구성된 한글 스크린에디터는 Input/output handler의 극히 일부분인 한글 Automata 부분만 조금 변경시키므로서 거의 모든 한글 터미널에서 사용이 가능하다. 그림 1은 한글 스크린에디터의 블록구조 및 데이터, 제어의 흐름을 나타내고 있다.

3. 한글 포매터의 구조

포매터는 워드프로세서의 한 부분으로 에디터에 의해 작성된 서류를 좀더 짜임새있고 보기좋은 형태의 서류로 변환시키는 소프트웨어이다. 포매터는 에디터와는 달리 대화형으로 처리되지 않기 때문에 비교적 그 구조도 간단하다. 포매터도 에디터와 마찬가지로 한글 터미널을 사용하여 수행이 될 수 있게 개발을 한 다음 Bit-map display 시스템과 연결하여 사용하게 한다.

한글 포매터의 구성은 크게 Main routine, Command handler, Text handler 및 Input/output handler로 나누어지며 Main routine은 disk로부터 읽은 서류를 명령줄과 text줄로 구분하여 Command handler와 Text handler로 제어시키며, 여기서는 각각의 Command에 대한 이름 table을 이용한다.

Command handler는 각각의 명령어와 관계가 되는 여러 flag 및 data structure의 값을 변경시키며, 각 명령어에 대한 default 값 table을 이용하여 default 값 처리도 해준다.

Text handler는 Main routine에서 읽은 한줄의 text를 formatting용 flag 및 status의 value를 이용하여 출력용 buffer에 저장시킨다. 여기서 출력용 buffer는 2 column 출력을 위하여 2차원 array로 구성되어 있다. 그리고 Input/output handler는 disk에 구성된 파일의 한줄을 읽어주는 routine과 출력 buffer의 내용을 디스크 혹은 프린터로 출력시켜주는 routine으로 구성되어 있다. 이 포매터도 I/O handler의 코드변환 routine을 변경시킴으로서 여러 한글 프린터에 쉽게 응용될 수 있게 설계되어 있다. 그림 2는 한글 포매터의 블록 구조를 나타내고 있다.

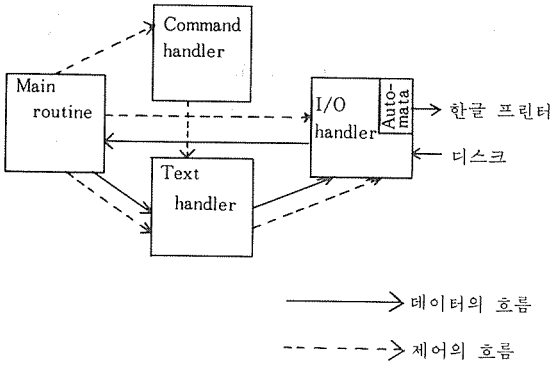


그림 2. 한글 포매터의 블록구조

4. 한글 Bit-map display system

제 2 장, 제 3 장에서 설명한 에디터 및 포매터는 최종적으로 한글 Bit-map display시스템과 연결되어 UNIX를 사용하는 사무용 워크스테이션에서 수행 가능하게 종합이 된다.

한글 Bit-map display시스템은 키보드에서 받은 코드를 automata를 거쳐 내부코드로 변환시켜주며, 내부코드를 받아서 Font file에서 지정된 한글 Font를 찾아서 CRT 스크린 상에 한글을 나타나게 해준다. 이때 제어코드의 처리 및 Escape 연속 code를 역시 처리해주며 한글의 글씨체 선택 및 글자의 Font크기 선택도 이 Bit-map display시스템에 의해 이루어진다. 기본적으로 한글은 24×24, 영문은 16×24의 크기를 가진다.

그림 3은 한글 Bit-map display 시스템의 블록구조를 나타낸다.

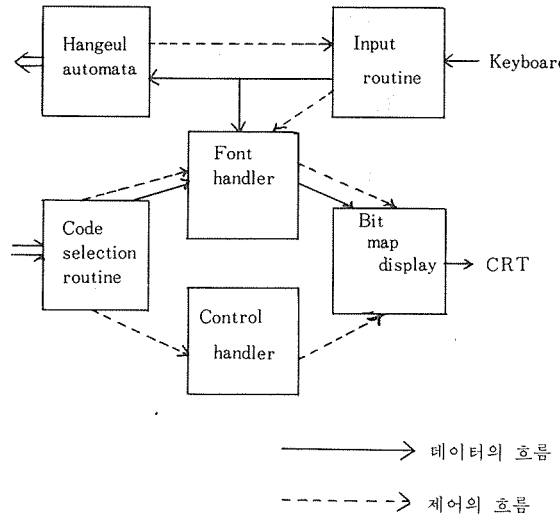


그림 3. 한글 Bit-map display 시스템의 블록구조

5. 한글 워드프로세싱 시스템의 종합화

2, 3, 4장에서는 한글 워드프로세서의 기본이 되는 에디터, 포매터 및 한글 Bit-map display system에 대해 설명하였다. 여기서는 이것들이 종합된 한글 워드프로세싱 시스템에 대하여 이야기한다. 기본적으로 한글 워드프로세싱 시스템은 에디터, 포매터의 기능은 물론 도움말 시스템 및 화일처리에 필요한 Utility 명령어 시스템 등이 추가가 되어 menu driven 시스템으로 운영이 된다. 도움말 시스템은 전체적인 한글 워드프로세서의 사용법 및 동작법에 대한 설명 및 스스로 한글 워드프로세서를 배울 수 있는 학습 menu로 구성이 되며 화일처리 Utility는 화일에 관계된 UNIX 명령어를 효율적으로 쉽게 불러서 연결시켜 주는 menu-driven으로 되어 있다.

실제 한글워드프로세싱 시스템의 수행시에는 사용자가 선택할 menu에 따라 여러개의 process가 존재한다.

6. 결론

본 논문에서는 에디터, 포매터, Bit-map display 시스템 등이 종합된 UNIX O.S용 한글 워드프로세싱 시스템에 대하여 이야기하였다.

이 시스템의 특징은 UNIX Machine에 쉽게 Porting되어 사용될 수 있다는 점과, 여러 한글 터미널을 사용할 수 있도록 쉽게 변환 가능하다는 점 및 한글의 display를 위하여 Bit-map graphics 방식으로도 Implement하였기 때문에 한글의 글자체, 글자크기의 변환을 쉽게 처리할 수 있다는 점 등이다.

앞으로 한자처리 및 graphics와 text의 종합적 처리 등의 과제가 남긴 하였지만 Bit-map display 방식의 잇점을 이용하여 쉽게 기능첨가가 가능할 것으로 본다.

참고 문헌

1) N. Meyrowitz, A.Van Dam, Interactive Editing System:Part Ⅰ, acm computing surveys, Vol 14 No. 3 Sept. 1982.
2) N. Meyrowitz, A. Van Dam, Interactive Editing System:Part Ⅱ, acm computing surveys, Vol 14 No. 3 Sept. 1982.
3) B. W. Kernighan, Software tools, Addison-wesley, 1976.
4) 오길록 외 36명: 사무자동화 시스템 개발에 관한 연구, 1984. 3. '83년도 과기처 특정 연구사업.

P.34에서 계속

大한 分野가 展開될 것이다.

美國에는 약 6,000만명의 知的 勞動者가 있으나 이 가운데 불과 10~15%의 사람만이 Perscom을 소유하고 있다. 우리들이 成長할 소지는 충분히 존재한다. 현재의 Perscom業界는 잠시 휴식을 취하고 있는 것이며 다시 그 위력을 발휘하게 될 것이다. 마라톤 레이스에 대비하기 위한 휴식에 불과한 것이다.

비지니스의 기본적인 요소에 있어 自己修練을 成就한다면 相互利益, 信賴, 誠実에 바탕을 둔 비지니스 관계를 구축한다면 바로 이때 우리들은 生存하게 될 것이며 繁榮하게 될 것이다.