Shop Floor Level의 Data공유화를 위한 공통 DB Interface API의 필요성과 그 역할

정성준* · 호리우찌* · 백일현* · 김태완*

Common DB Interface APIs for Data Sharing at Shop Floor Level.

Sung-Jun Chung · Horiuchi · Il-Hyun Baek · Tae-Wan Kim

〈Abstract〉

본 연구는 생산 현장과 기술 설계 부서를 효율적으로 연결시키고 기술 자료를 공유화하기 위해서는 각 역할분담(Software Module)에서 공통DB(EDB/MDB)를 공유화할 뿐만 아니라, 이러한 공통 DB를 특별한 지식이 없이 이용할 수 있는 공통 API(Application Program Interface)와 DB를 통합적으로 관리해주는 Server Process의 필요성을 업계의 동향과 국제 표준화 동향을 참고하여 설명하고자 한다.

주요어 : DBMS, Shop Floor, FMS, API, Network, GUI, MAPLE, CALS

1. 서론

근래 미국, 일본을 비롯한 세계 곳곳에서 경영혁신운동(Business Process Reengineering)이 활발하게 진행되고 있다. 국내에서도 경영혁신운동을 도입하여 큰 성과를 얻고 있는 기업체도 적지 않다. 그러나 미국의 경우에도 이러한 경영혁신운동의 70%가 실패하고 있다고 한다. 큰 투자를 해서 추진했는데 기대한 것과 달리 크게 업적을 올릴 수 없거나, 오히려 업적이 저하된 업체도 있다. 또 성공 사례로 평가를 받고 있는 기업체에도 과다한 해고로 인한 고용문제, 계속적인 Software의 개발/보수에 인한 Software의 Back Log 문제 등 경영혁신운동의 부작용으로 새로운 문제가 발생하고 있다. 그러면 이 경영혁신운동을 기술면에서 지원하고 있는 정보관리 기술을 중심으로 어디에 문제가 있는지 분석해보고 이에 대한 대책안을 제시해 보도록 하겠다.

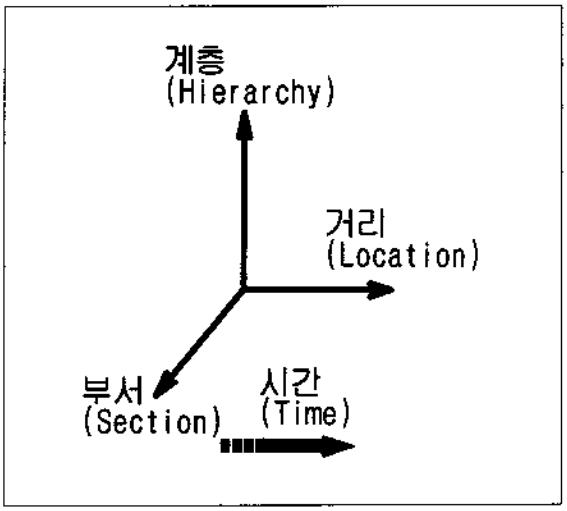
2. 공통 DB Interface API의 필요성

2.1 정보 공유화의 장벽

Shop Floor Level뿐만 아니라 크게는 CIM에 이르기까지 가장 핵심이 되는 요소는 '정보'의 효율적인 운영이라고 할 수 있다. 그러나 필요한 정보를 적시에 이용하는 것이 결코 쉬운 일이 아니다. 정보의 이용자인 '나'(사용자 또는 생산현장/설계현장의 각 Process)와 얻고자하는 '정보'사이에는 여러 가지의 장벽이 있다. 하나는 시간적인 장벽이며 또 하나는 공간적인 장벽이다. 시간적인 장벽이라 함은 '내'가 필요한 정보

* 통일중공업(주) 기술개발중앙연구소

가 이미 과거의 것으로 현재 존재하지 않음을 말한다. 이 장벽을 극복하려고 하면 그때 그때 필요한 정보를 기록하여 막대한 과거의 정보를 저장,활용하는 수단이 필요한다. 그리고 공간적인 장벽이라 함은 정보가 현재 어딘가에 존재하고 있는데 자기 능력으로 취득할 수 없음을 말한다. 이 공간적인 장벽도 세 가지의 방향이 있다(〈그림 1〉 참조). 하나는 거리상의 문제이고, 다른 하나는 부서간이나 업무상의 관계에 생기는 문제이며, 마지막 하나는 계층구조에서 생기는 문제라고 할 수 있다. 첫째, 거리(Location)상의 문제라는 것은 '내'가 있는 장소나 사용하고 있는 장비에서 정보가 격리되어 있어서 이용 못하는 경우이며, 이것을 해결하려고 하면 정보가 있는 장소(Data Source)와 이용자가 있는 장소(User Environment)를 연결시켜주는 방법 및 수단이 필요하다. 둘째, 부서 및 업무간의 문제라는 것은 특정 부서 및 업무에서 관리되고 있는 정보를 다른 부서에서 이용 못하는 경우를 말한다. 원래 정보는 특정 부서에서 관리/통제되는 것이 일반적인데, 다른 부서에서 꼭 필요한 정보를 얻기까지는 많은 통제가 따르고, 또 결재 과정에서 상당한 시간이 요구되거나 이용하는 방법이 제공되지 않는 경우도 많이 발생한다. 또 나아가서 자기 부서의 업무 편의를 위해 일부러 정보를 제공하지 않는 부서이익주의(Sectionalism)도 큰 장애라 할 수 있다. 반면에 타부서에 많은 협조를 하려해도 부서간 전산화의 진도 차이로 인해 정보의 흐름에 Gap이 생겨서 수월한 정보교환이 어려운 경우도 있다. 이러한 조직/사회적인 문제를 해결하려고 하면, 전사적인 Top-Down방식의 Process개선과 그것을 지원하는 정보의 구조적 분석 및 간편한 정보 접근 방식(Access Method)이 필요하게 된다. 특히 전사적으로 공통으로 이용하려고 하면 특별한 지식이나 교육이 없이 즉시 사용할 수 있는 월등한 User Interface가 요구된다. 셋째로 계층적인 문제라는 것은 상위 계층에 있는 관리자들이 현장에 있는 정확한 정보를 파악할 수 없는 경우나, 반대로 실무를 담당하고 있는 실무자들이 회사의 방향성이나 사정 등을 이해하지 못하고 형식적으로 업무를 수행하는 것을 말한다. 또 실무자가 정보를 올렸을때 결정 과정에서 과다하게 시간이 소요돼서 기회를 잃거나, 경영층/관리층에서 직접 정보수집(정보 단말기의 조작이나 부하직원의 의견수렴)을 하지 않고 가공된 정보나 보고만 들어서 다각적인 시각에서 판단을 내릴 수 없는 경우도 이에 속한다. 이러한 문제들을 해결하려고 하면 효과적인 의사 전달 방법과 다양한 방법으로 정보를 처리하는 수단이 필요하게 될 것이다. 그러한 시간적/공간적인 4차원의 장벽을 극복하기 위해서 Database, Network, 그리고 GUI의 요소 기술들을 효과적으로 이용해서 시스템을 구축해야 된다.



〈그림 1〉 4차원의 장벽

2.2 표준 API

DB, Network, GUI를 이용해서 User Application을 개발하려고 하면, 각각 표준적인 API를 이용하는것이 가장 효율적이다. API도 Platform, Vendor, Vendor Group에 따라 서로 호환이 되지 않는 경우가 있어 사용자는 각 특성을 충분히 이해하여 이용할 필요가 있다. Unix와 Windows 3.1의 두 가지의 Platform에 대한 API 표준현황은 〈표 1〉과 같다.

2.3 FMS운영 EDB/MDB Interface API

상술한 바와 같이 분산형 시스템 Application을 개발할 때 DB, Network, GUI의 각 API를 이용하게 되

〈표 1〉 표준 RDBMS, Network, GUI API

RDBMS Interface API	중심기관	Unix	Windows 3.1	비 고
ODBC	Microsoft		O	Windows 표준
CLI	X/Open	O		
Vendor API (Oracle PRO*C, OCI 등)		O	O	
RIS	Intergraph	O	O	Unix/Windows/Windows NT
Network API				
Low Level API				
Berkeley Socket Library		O		Unix 표준
Winsock	Microsoft		O	Berkeley Socket과 기능상 호환
TLI		O		System V 표준
High Level API				
ONC+ (RPC)	SUN 외	O		
DCE (RPC)	OSF	O		
Object Management				
CORBA	OMG	O		
COM/OLE2	Microsoft DEC	O	O	CORBA와 상호호환성을 추진중
GUI API				
X-Window/Motif	OSF	O	△	Wintif(Windows판 Motif) 개발중
Windows API (Win16, Win32)	Microsoft		O	
VBX(Visual Basic Custom Control)	Microsoft		O	VB,C++등 지원
TCL/TK	PDS	O	O	확장 가능한 Interpreter 언어

는데 DB Interface API는 SQL(Structured Query Language: DB조회 언어)의 지식을 전제조건으로 하고 있고, Network API를 습득하는데 상당한 시간이 필요하게된다. 그러한 부담을 각 Module마다 가져야 된다면 전체 시스템 개발 차원에서 보면 상당히 비효율적이다. 그래서 Shop Floor Control에서 이용되는 각 Application Program에서 쉽게 이용할 수 있도록 FMS 공통DB(EDB/MDB)의 Interface API를 제공하여 일관성있는 Data관리를 실현해야 한다. 각 Module별로 개발된 대표적인 API는 다음과 같다.

2.4 Implementation

EDB/MDB Interface API를 Implementation하는데 몇가지 방법이 있다. 여기서는 RDBMS Interface API와 Network API, 그리고 Object Management Technology를 이용한 구현방법에 대해서 알아보겠다. 어느 방법을 이용해도 User Application쪽에서 보면 동일한 Interface로 이용할 수 있다.

Case 1: Application마다 각자 RDBMS Interface API를 이용해서 EDB/MDB Interface API 구현

가장 일반적으로 사용하는 방법인데 현재로서 가장 안정성 있고 현실적인 방법이라고 할 수 있지만 DB

공통

Item정보 조회	get—item—info	
공정계획정보 조회	get—proc—list	

공정계획

공정계획정보 등록	set—proc—list	
개별공정정보 등록	set—proc—info	
개별공정정보 조회	get—proc—info	
선반공구Holder 계승List 조회	get—lathe—holder—class—list	
선반공구Holder정보 조회	get—lathe—holder—info—by—class	
크기,능력별로 가공기계 검색	search—machine—by—size—and—power	
가공기계 계승List 조회	get—machine—class—list	
가공기계정보 조회	get—machine—info—by—class	

생산계획

제조 Order 등록	add—prod—order	
제조 Order 조회	get—prod—order	
제조 Lot Order 등록	add—prod—lot—order	
제조 Lot Order 조회	get—prod—lot—order	
공정전개정보 등록	add—proc—expansion	
공정전개정보 조회	get—proc—expansion	
순서계획 등록	add—dispatched—plan	
순서계획 조회	get—dispatched—plan	

공정제어/관리

가공Program을 Local로 복사	copy—part—program—to—local	Item, 공정별로
기계별 공정정보 참조	get—dispatched—plan	
특정Pallet위치정보/상태 참조	get—pallet—location	
특정Pallet위치정보/상태 변경	set—pallet—location	
전체 Pallet Stocker정보 참조	get—pallet—stocker—info	

시스템 상태감시

공정별 가동시간 등록	add—running—time—in—process	공정마다
공정별 절삭시간 등록	add—cutting—time—in—process	공정마다
기계별 작업중단시간 등록	add—down—time—by—machine	
기계별 작업시간 등록	add—working—time—by—machine	

Setup 정보관리

공정별 필요기구번호 조회	get—tool—list—in—proc	
공정별 Setup순서정보 조회	get—setup—info—in—proc	
공정별 Setup Graphic정보조회	get—setup—image—file	
양/불량품 정보 등록	set—item—pass	PASS/NG

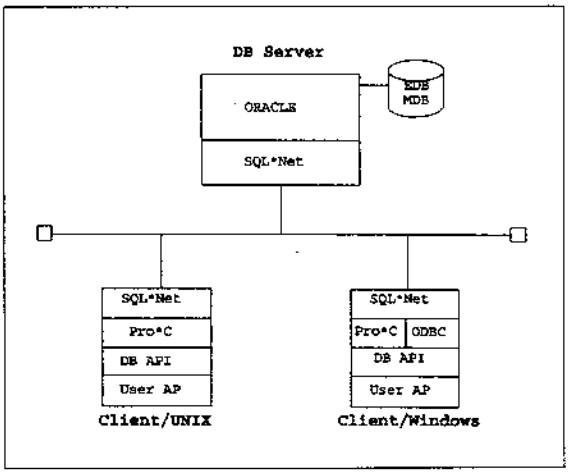
실적관리

기계별 월/주/일별 가동시간조회	get—running—time—by—machine	
기계별 월/주/일별 작업시간조회	get—working—time—by—machine	
기계별 월/주/일별 절삭시간조회	get—cutting—time—by—machine	
기계별 월/주/일별 중단시간조회	get—down—time—by—machine	
가공대상 Item번호 조회	get—planned—item—list	

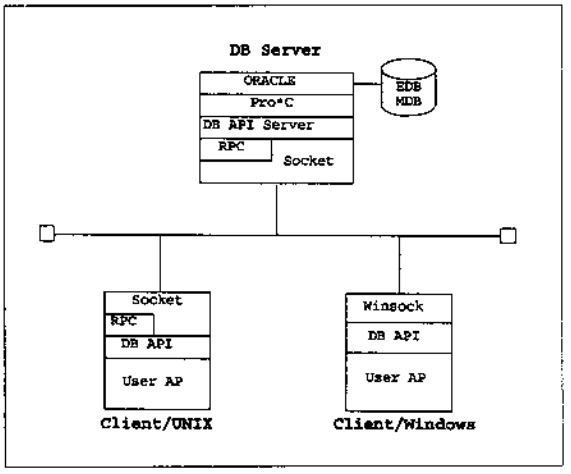
Vendor에 의존성이 크고 각 Client Machine마다 Run Time Module을 Install해야되므로 비용 부담이 크다 (〈그림 2〉 참조).

Case 2: Network API를 이용해서 EDB/MDB Interface I/F API 구현

이 방법은 Client쪽에서는 DB에 관한 처리를 전혀 하지 않고 Server쪽에 DB Access Request Message만 Network API를 통해서 보내준다. 이 Message를 받은 DB API Server Process는 Message의 내용을 해석하여 적당한 SQL문장을 생성해서 DB에서 필요한 Data를



〈그림 2〉 RDBMS Interface API에 의한 Implementation

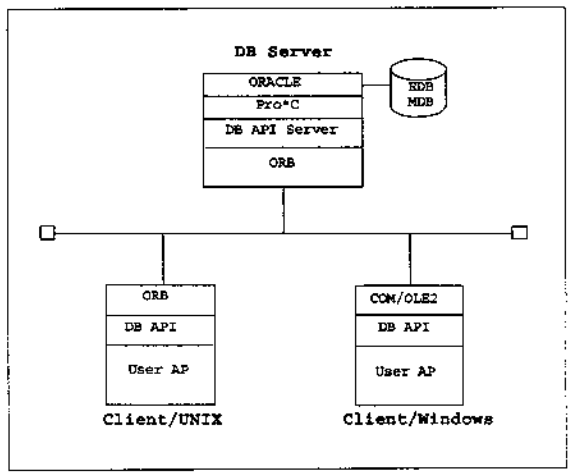


〈그림 3〉 Network API에 의한 Implementation

가지고 온다. 그리고 이 결과를 다시 Network API를 통해 Client쪽으로 Message로 보내준다(〈그림 3〉 참조). Case 1의 시스템 구성에 비해 저렴한 구성이 가능하게 된다. 이러한 Network API를 이용하는 경우도 Socket Library를 이용하는 방법과 DCE, ONC+ 등의 RPC(Remote Procedure Call)를 이용하는 방법이 있다. Socket는 Unix에서 표준으로 이용하고 있는 Library이고 호환성에 문제가 없고 Windows에서도 "Winsock"라는 API가 거의 표준으로 채택되고 있다. 그러나 RPC보다 추상화 Level이 낮고 작업량이 많아진다. RPC는 Socket에 비해서 다양한 기능을 지원하며 사용자에게 편리함을 제공하고 있으나, 같은 Unix내에도 Vendor에 따라 OSF/DCE와 Sun/Solaris를 비롯한 ONC+의 서로 호환성이 없는 표준이 존재하며, Windows에서는 표준이라고 할 수 있는 API가 없다.

Case 3: Object Management Technology를 이용해서 EDB/MDB Interface I/F API 구현

이 방법은 차세대 API인 Object관리기법(Object Management Technology)을 이용하면서 Client와 Server를 Object Level로 연결시키고 Network을 의식안해도 DB API를 작성할 수 있다. 가장 대표적인 표준 API는 Unix에서는 OMG(Object Management Group)의 CORBA(Common Object Request Broker Architecture)이며, Windows에서는 COM(Component Object Model)/OLE2(Object Link and Embedded-2)다(〈그림 4〉 참조). 이 CORBA와 COM/OLE2는 상호 공유화(Interworking)할 수 있도록 연구가 진행 중이다. 지금은 Unix와 Windows를 동시에 지원하는 시스템이 완성되어 있지 않아서 이용하기 어려우나 차세대 Application에서는 상당히 보급될 것으로 전망된다.



〈그림 4〉 Object Management Technology에 의한 Implementation

3. 표준화 동향

본 연구와 관련되는 표준화의 동향에 대해서 알아보겠다. FMS, Shop Floor Control 등의 정보관리와 관련해서 MMS, STEP, MANDATE, MAPLE, CALS 등

의 표준화 작업이 진행 중이고, 그 중의 MAPLE과 CALS는 본 연구를 진행하는 데 큰 도움을 주었다. 이하에 MAPLE과 CALS에 대해서 간략히 설명하고자 한다.

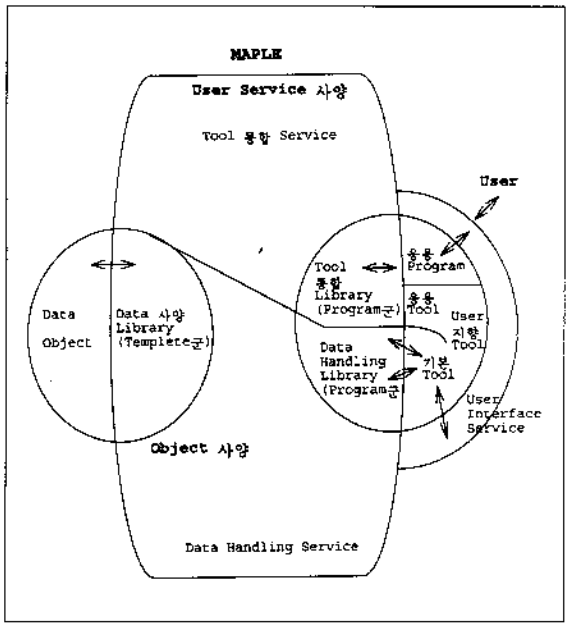
3.1 MAPLE (ISO / TC 184 / SC 5 / WG 4)

MAPLE(Manufacturing Automation Programming Language Environment: A common support facility for multiple, independent programming language for manufacturing devices and controls-overview : ISO/TR 12186)은 제조Cell을 제어/관리하는 Program을 작성하는 Programmer를 지원하는 Programming환경의 표준이다.

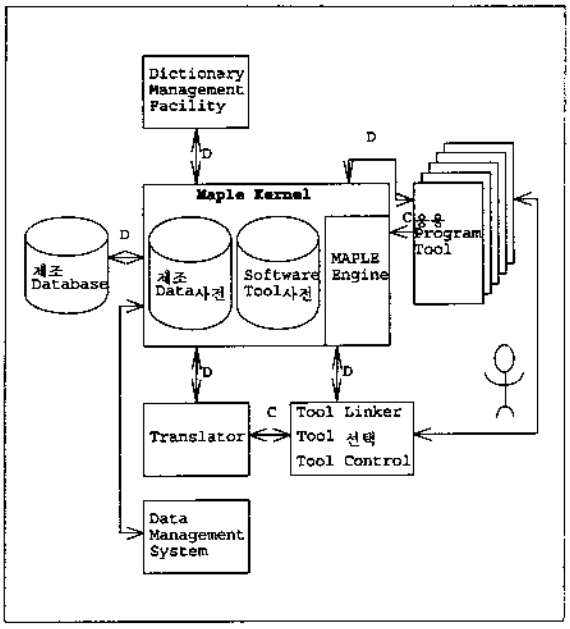
MAPLE의 개념

1. 제조 현장의 생산Data(일정계획, 공정설계, 작업설계, 공정제어/관리에서 이용하는 Data)를Handling해서 관련되는 응용Software를 통합 운영할 수 있도록 User Interface Service, Tool 통합 Service, Data Handling Service 등의 각종 Service을 제공한다(〈그림 5,6〉 참조).

2. 제조Cell을 추상화된 가상Cell로 정의하여, System Integrator(SI)는 이 모델 속에서 Data 및 제품의 흐름을 Control해서 자동제조를 수행할 수 있도록 Application Program을 개발해야 된다. 이 과정에서 특정한 제조시스템에 대해서 전 Task를 분석, 구조화하고, 각 Sub-Task의 사양을 명확화한다. 그리고 제조시스템에서 제조요구는 분석된 상세한 요구로 변환되므로, 효율적으로 Cell System의 자원을 이용하는 Scheduling이 가능하게 된다. 그러한 Application Program의 사양을 명확화시키고 가상Cell System을 통합하는 SI를 지원하는 것이 MAPLE이다.



〈그림 5〉 MAPLE Architecture

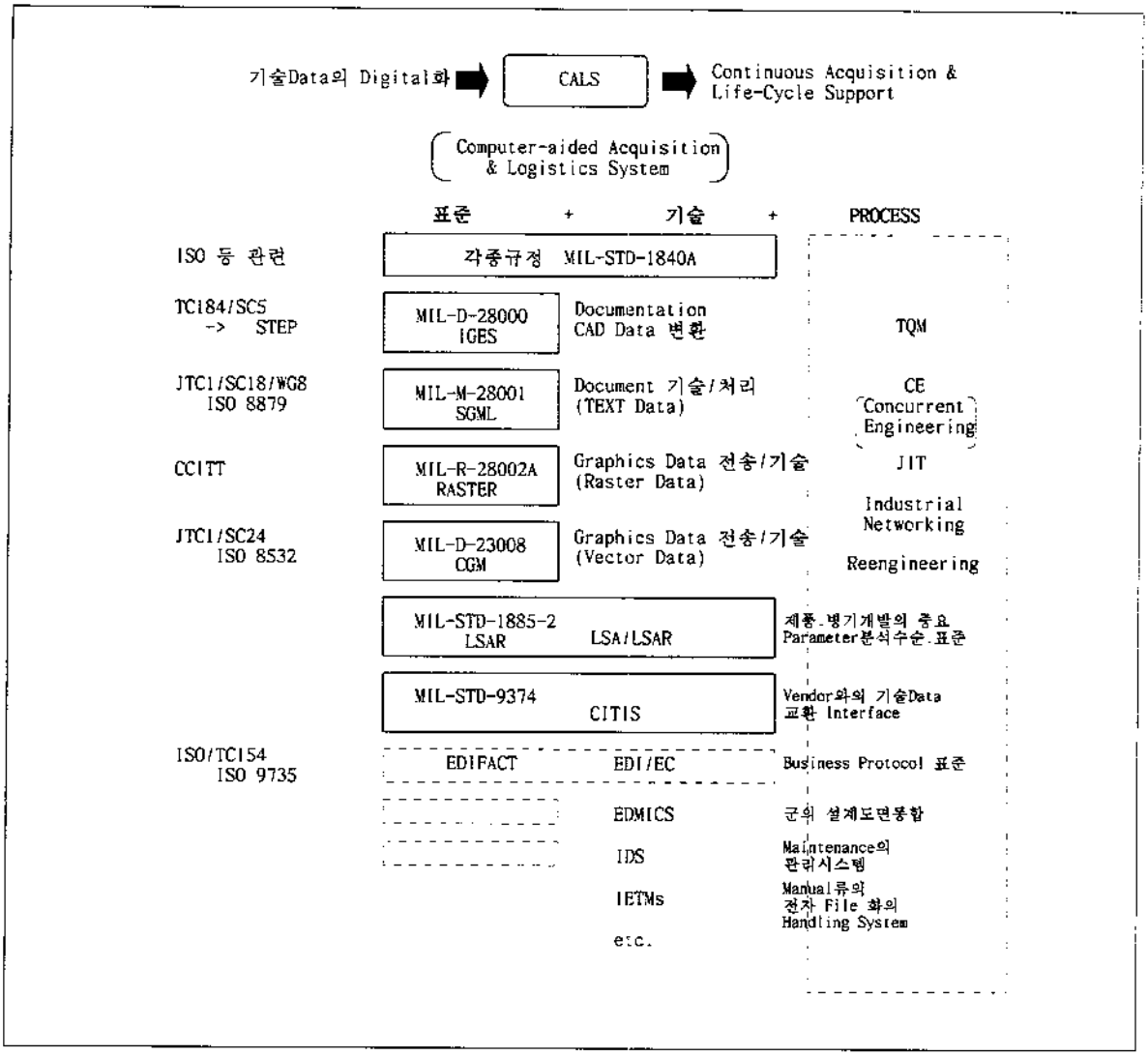


〈그림 6〉 MAPLE Architecture - Sketch

3.2 CALS

CALS(Computer-aided Acquisition and Logistics Support)는 미국 Military Logistics사에서 시작해서 미국 국방부(DOD)를 비롯해서 정부기관, 기업체로 확산되고 있는 합리화 혁신운동이다.

CALS는 MIL-HDBK-59E(draft:18 June, 1993)에 의하면 "방위시스템의 운영을 종이를 중심으로 한



〈그림 7〉 CALS의 전체개념도

Manual에서, 통합되고 고도로 자동화된 조달/지원 Process로 이행하는 것이다. 그러한 자동화, 통합화된 Process가 지향하는 것은 IWSDB(Integrated Weapon System Database)이며, 이 개념은 분산된 Data를 하나의 논리Database로 통합하는 것이다. 설계, 제조, 지원 Process의 자동화와 통합은 이 Process를 간소화하여, 방위시스템의 신뢰성과 품질을 향상시키고, 동시에 Life Cycle Cost를 절감시킨다."라고 되어 있다.

현재는 이 운동이 확산되어 CALS자체가 Vendor와 기술Data를 공유화하기 위한 기술 및 시스템개발적인 측면, STEP, SGML 등 국제표준을 포함시키는 표준화적인 측면, 그리고 TQM(Total Quality Management), JIT(Just-in-Time), CE(Concurrent Engineering), Reengineering 등 생산/판매/물류/기술의 Process혁신 활동적인 측면의 각각 세 가지의 역할을 가지는 포괄적인 개념으로 변화하고 있다(〈그림 7〉 참조).

4.관련 기술의 전망

근래의 Business Process Reengineering은 Software가

술을 동원해서 생산성을 향상시키는 사례가 많다. 그것은 기업, 사회가 복잡화됨에 따라 Software 통합의 수요가 많아지고 있다는 것을 의미한다. DB, Network, 그리고 GUI의 요소 기술은 사용자의 생산성을 증대시키는데 큰 도움을 줄 수 있다. 그러나 Application Program을 개발하는 입장에서는 그런 기술들을 습득하는데 새로운 부담으로 인식되고 있는것도 사실이다. 특히 GUI Base의 Application개발은 필수적이나 전통적인 Application에 비해 상당한 시간 투자가 요구된다. 이를 극복하려고 California대학 Berkeley교의 Dr. John K. Ousterout(현:Sun Microsystems)은 Tcl/Tk를 개발하여 Public Domain Software로 보급하고 있다. Tcl(발음:"tickle")은 "Tool Command Language"를 의미하여 Unix Shell과 비슷한 Command Interpreter언어다. Lisp의 개념을 포함하고있으며 확장성이 뛰어나다. Tk는 Tcl을 X-Window환경에서 이용할 수 있도록 확장한 "Tool Kit"이며 X의 widget들을 Control할 수 있다. 이 Tcl/Tk를 Network Programming Tool로 확장시키고 RPC를 가능하게 만든 Tcl-DP나 Oracle, Sybase 등과 연결해서 DB Client Tool로 이용하는 oratcl, sybtcl 등이 있다. 이러한 자유도와 확장성이 풍부한 Application Builder를 이용해서 짧은 시간내에 목적으로 하는 Application을 개발하는 것이 바람직하다.

5. 결론

본 논문에서는 Shop Floor Level에서 생산정보를 관리하기 위한 각종 Application을 개발할 때, 공통으로 이용할 수 있는 DB Schema(EDB/MDB)와 효율적으로 Application을 개발할 수 있는 DB Interface API가 중요하다고 논하였다. 그러한 API를 Implementation할 때 업계에서 표준으로 이용하고 있는 각종 API를 사용해서 개방적인 시스템을 개발하는것이 중요하다.

【참고문헌】

[1] Object Management Group, "The Common Object Request Broker: Architecture and Specification", OMG Document Number 91.12.1.

[2] Microsoft, "Microsoft ODBC Interface - Application Programmer's Guide".

[3] Microsoft, "Microsoft ODBC Interface - API Reference".

[4] Microsoft, "Windows Sockets An Open Interface for Network Programming under Microsoft Windows".

[5] John Bloomer, "Power Programming with RPC", O'Reilly & Associates, Inc.

[6] W.Richard Stevens, "UNIX Network Programming", Prentice Hall.

[7] "분산형 컴퓨팅行 열차를 잡아라", 오픈컴퓨팅, 1995. 3

[8] Karen Watterson, "VISUAL BASIC Database Programming", Addition Wesley.

[9] John K. Ousterhout, "Tcl and the TK Toolkit", Addison Wesley.

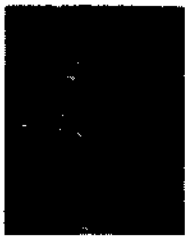
[10] 日本規格協會, "JISハンドブック ロボット・FAシステム-1994", "參考: [FAシステム] 5.FAの標準化關連技術情報"

[11] Michael Hammer and James Champy, "REENGINEERING THE CORPORATION : A Manifesto for Business Revolution".

정성준
부산 동아대학교 전자공학과 전공
91. 경영기술지도사 자격획득.
협력업체 통합 전산망 구축
세일 MRP 시스템 개발
인사관리 시스템 개발
사내 CAD/CAM 표준화 위원회 위원장
사내 CIM 추진위원회 주간 간사
현재 통일중공업 정보시스템부 부장, 기획조정부 부장 겸임.

호리우찌 겐이찌
1960년생.
통일중공업(주) 정보시스템부
쓰쿠바 대학에서 물리학을 전공하였으며 Database, Network 및 GUI에 관심을 가지고 있음.

김태완
부산외국어대학교에서 컴퓨터공학을 전공했으며, Network Programming 및 OOP분야에 많은 관심을 가지고 있음.



백일현
부산외국어대학교에서 컴퓨터공학을 전공했으며, Manufacturing Database를 비롯한 기술자료관리 Database에 관심을 가지고 있음.