CIM사업 연구기획 및 국제표준기술 이전

Planning of CIM Program and Transfer of
International Standards

연구기관
한국과학기술연구원

과학기술처

제출문

과학기술처장관 귀하

본 보고서를 "CIM사업연구기획 및 국제표준기술이전" 과제의 최종보고서로 제출합니다.

1992년 2월

주관연구기관명 : 한국과학기술연구원
과제연구책임자 : 강 무진(KIST, CAD/CAM연구실)
연구원 : 고 희동(KIST, CAD/CAM연구실)
박 면웅(KIST, CAD/CAM연구실)
신 숙재(KIST, CAD/CAM연구실)

요 약 문

I. 연구제목

CIM사업 연구기획 및 국제표준기술 이전

II. 연구의 목적 및 중요성

오늘날, 국가의 경쟁력은 그 제조업의 경쟁력에 의해 결정된다고 해도 과언이 아니다. 경쟁력을 결정하는 요인은 金利와 같은 企業外的인 요소와, 품질, 납기, 가격 등 企業內的인 것이 있겠으나, 제품을 값싸고 좋게 빨리 만들어 市場에 공급함으로써 경쟁력을 확보한다는 것은 企業內的인 面에서 절대적인 명제라 할 수 있다. 이를 위하여 설계기술(CAD/CAE/CAPP), 제조기술(CAM), 측정.검사기술(CAQ), 생산관리기술(MRP)등 생산의 제반 요소기능들을 통합하고자 하는 노력(CIM)은 현대 생산기술의 주된 관심사이며, 선진 각국에서도 CIM기술의 개발을 위하여 막대한 노력을 기울이고 있다.

미국의 경우 공군에서 주관한 ICAM(Integrated Computer Aided Manufacturing)사업과 NIST에서 수행한 AMRF(Automated Manufacturing Research Facility) 사업, 그리고 과학 재단(NSF)에서 지원하는 ERC 계획들을 통하여 생산시스템의 知能化, 統合化, 標準化 技術 확보를 추진해 왔다. 유럽에서는 단일 경제 블록 탄생을 앞두고 이미 1980年代 初부터 ESPRIT, EUREKA, BRITE/EURAM, RACE 등의 대형 프로그램을 결성하여 정보처리, CAD/CAM 및 CIM, 센서, 레이저, 통신, 로보트 기술들을 집약

한 생산시스템 개발을 범유럽적으로 추진하고 있고, 발생하는 知的 所有權은 개발팀의 소유로 하되 EC 내에서 또는 EC를 위하여 사용하는 것을 원칙으로 하고 있다. 日本은 IMS라는 三極 프로그램을 제안하여, 미국, EC와 함께 차세대 생산시스템 기술의 체계화, 표준화를 주도하고자 하고 있다. 아직 사업 형성 단계에 있는 IMS 프로그램이 本格的으로 수행되고 그 硏究開發 結果가 可視化되기 시작하면, 거의 모든 생산 기술이 知的所有權에 묶여 우리나라를 비롯한 非參加國들은 엄청난 技術的, 經濟的 從屬을 감수해야 하는 심각한 상황이 예상된다.

이와 같이 先進國들의 기술 개발 동향과 지적 소유권 Issue의 추세를 볼 때, 우리도 이 問題의 심각성을 正視하고 능동적으로 대처해야 할 時點에 와 있는 것 같다. 본 연구는, CIM기술의 국내기반을 확보하고 선진국에의 기술종속을 탈피하기 위하여, CIM기술의 세계적 수준 및 발전동향과 각국의 CIM기술개발 사례를 면밀히 분석하고, CIM의 확산을 위한 국제적인 표준화 노력을 추적하여 국내에 보급하는 것을 목적으로 하고 있다.

III. 연구의 내용

본 연구에서 수행한 내용은 다음과 같다.

가) CIM기술의 세계적 수준 및 발전동향 분석

- 요소기술분야별 추이
- 각국의 CIM기술개발 전략 (사례연구)

나) 국제표준화 Activity에 가입, 국제표준연구의 현황조사 및 참여

SUMMARY

I. Title of the Project

Planning of CIM Program and Transfer of International Standards

II. Goals and Significance of the Project

The competitiveness of a nation is strongly influenced by the competitiveness of its manufacturers. One of the most important factors determining the competitiveness is how fast they can produce the products of good quality at the low price. The efforts to achieve this goal are recently focussed on integrating the whole activities of manufacturing, i.e. CAD, CAE, CAPP, CAM, and MRP.

In this point of view, most developed countries have carried out lots of projects aimed at achieving the CIM technology. For instance, European Community built several large-scale R & D program such as ESPRIT, EUREKA, BRITE/EURAM, and RACE in order to activate the potentials in information technology, CAD/CAM, CIM, sensors, lasers, communication, and robotics. It is expected that the results of these programs belong to EC and may be applied to their own usage only. In the United States, ICAM of the American airforces, AMRF and several ERC's of NSF are good examples of driving R & D in CIM fields. The principal aims of these activities are intelligence, integration, and standardization of manufacturing systems. Japan

has recently proposed the IMS initiative, so called three-pole program, in which it plans to lead the world-wide systemizing and standardizing activities in cooperation with USA and EC. It is anticipated that countries not taking part in IMS program - also Korea - will come to the hard situation, bacause most of the manufacturing technology will be regulated according to the knowledge right.

In this aspect mentioned above, we also have to recognize this problem seriously and should take the appropriate action in order to achieve our own technology and to avoid the technological dependency on the developed countries. The goal of this project is to survey the state-of-the-art and trends of CIM technology world-wide, to analyze the cases of CIM technology development, and to follow up the international standardization efforts in industrial automation field.

III. The Scope and Contents of the Project

The scope and contents of the project are as following :

(i) Analysis of the state-of-the-art and developing trends in CIM technology

- Components of CIM
- Case studies of CIM technology developments in each country

(ii) Participation in international activities on standardization in CIM technology field

- Survey the international standards
- Diffusion of the standards into Korean industry

Contents

Chapter 1 Backgrounds 11

Chapter 2 State-of-the-art and Trends of CIM Technology 13

Section 1 General remarks 13

Section 2 Design 15

Section 3 Fabrication 19

Section 4 Production control 22

Section 5 Networking and communication 23

Chapter 3 World Survey of CIM Technology Development 26

Section 1 Introduction 26

Section 2 Europe 28

Section 3 U.S.A 36

Section 4 Japan 45

Section 5 Developing countries 53

Chapter 4 International Standards in CIM 60

Section 1 Introduction 60

Section 2 ISO/TC 184 62

Section 3 Situation in Korea 69

References 71

Appendices 75

(i) Framework for CIM system integration

(ii) Reference model for shop floor production standards

(iii) Status of PDES-related activities

(iv) Companion standard for NC

목 차

제1장 서 론 11

제2장 CIM기술의 세계적 수준 및 발전동향 13

제1절 총론 13

제2절 설계기술 15

제3절 제조기술 19

제4절 생산통제기술 22

제5절 Networking 및 Communication 기술 23

제3장 세계각국의 CIM기술개발 사례 26

제1절 개요 26

제2절 유럽 28

제3절 미국 36

제4절 일본 45

제5절 개발도상국 53

제4장 CIM분야의 국제 표준 60

제1절 현황 60

제2절 ISO/TC 184 62

제3절 국내동향 69

참고문헌 71

부록 75

(1) Framework for CIM system integration

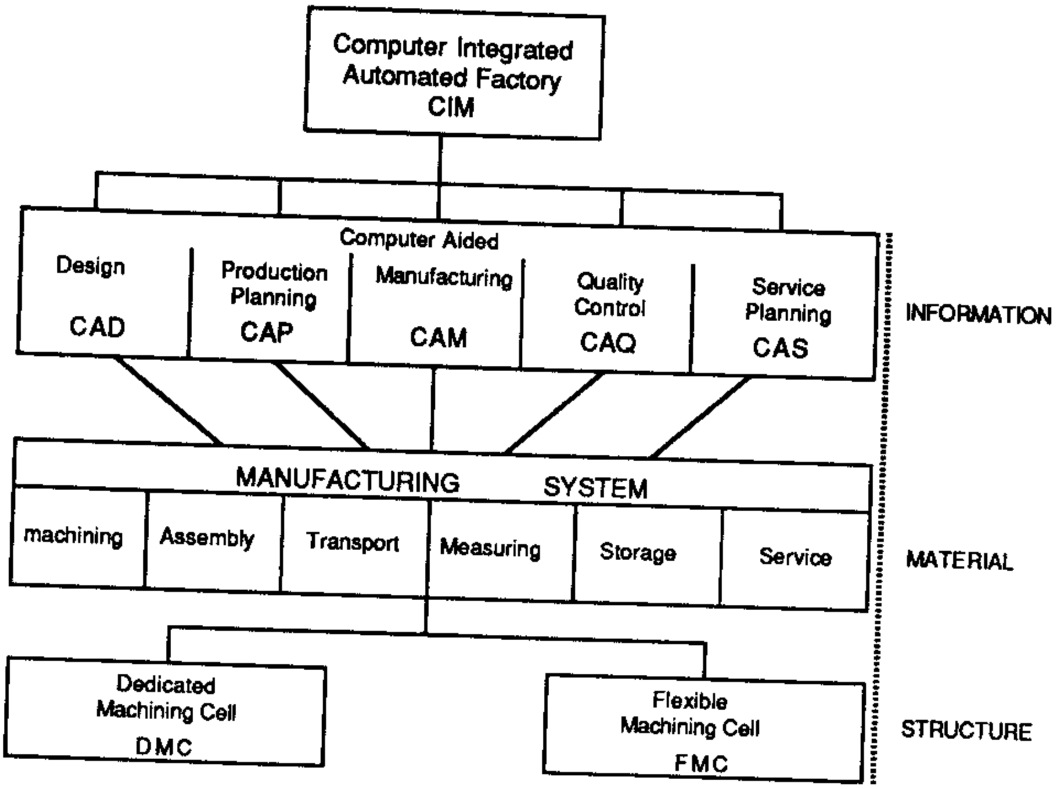
(2) Reference model for shop floor production standards

(3) Status of PDES-related activities

(4) Companion standard for NC

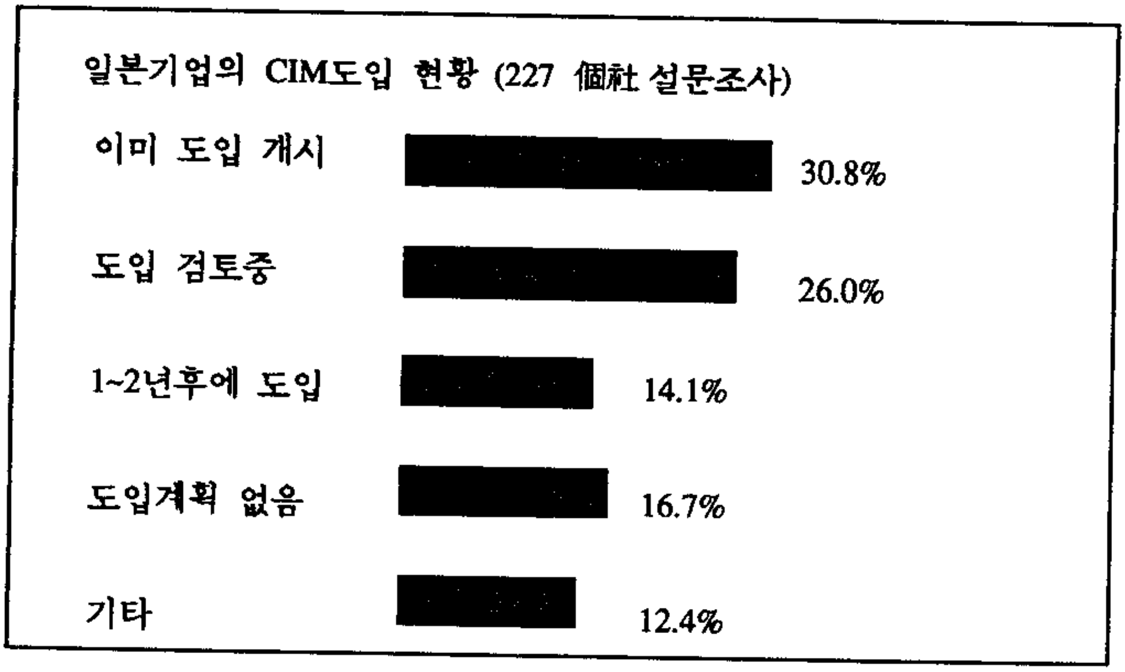
제1장 서 론

급변하는 세계시장 상황의 변화에 제조업이 능동적으로 대처하기 위한 방안으로서 전산기 원용(Computer Aided)기술은 현대 생산기술 관련 연구개발의 주 관심사가 되어 왔다. 생산의 각 부문에서 CAD,CAPP, CAM, CAQ, MRP 등 소위 CAx 기술이 꽃을 피우고 있는 가운데, 최근에는 각 부문에서의 유리된 자동화/전산화 단계를 뛰어넘어 제반생산 부문들 간의 정보교환을 일원화하여 全社的인 最適化를 추구하는 통합생산자동화(CIM)기술이 기업생존전략의 하나로 부각되고 있다. (도1참조)



<도1> CIM공장의 기능적 구조 및 CAx 기술의 적용
(Source : G. Spur)

일본의 경우 1988년 현재, 선단기업의 30.8%가 이미 CIM도입을 시작하였고 26%가 CIM도입을 검토하고 있는 실정임을 감안할 때(도2 참조), 선진제국과 경쟁해야 하는 우리나라 역시 기업생존의 차원에서 CIM 기술에 관심을 갖고 그 도입을 검토하게 됨은 당연한 귀결이라 할 수 있다. 그러나, 국내기업의 경우 CIM관련 인력과 정보가 부족하여 그 효율적인 추진이 이루어지지 못하고 있음은 안타까운 일이다.



<도2> 일본의 CIM도입 상황 (Source : 日本能率協會)

따라서, CIM국책연구개발사업의 일환으로 CIM에 있어서 세계적인 기술수준 및 발전추세를 분석하고, 標準化와 知能化를 추구하고 있는 國際機構의 활동을 추적하여 조속하게 국내 수요자들에게 전달하는 것이 필요하다.

제2장 CIM기술의 세계적 수준 및 발전동향

제1절 총론

세계의 생산기술을 주도하는 Initiative를 미국, 유럽, 일본의 三極으로 볼 때, 이들의 CIM기술개발 전략 및 기술수준은 각각 독특한 특색을 가지고 있다.

미국은 전통적으로 막강한 기술력을 가지고 있는 컴퓨터 회사를 중심으로, 시스템 통합을 위한 Architecture를 정의하고 그 Slot에 맞는 요소제품을 상품화하는, 전형적인 Top-Down 방식이 보편화되어 있다. 그리고, GM자동차의 MAP Initiative에서 볼 수 있듯이 세계의 표준화 방향을 선도하기 위한 노력에 큰 힘을 쏟고 있다. CIM의 실현에 있어서는, <도3>에 표시된 바와 같이 GT기술을 통합의 핵심수단으로 삼아 Pilot plant들을 구축, 테스트하는 접근 방법을 취하고 있다. 기술적으로는 OA 분야에 매우 앞서 있으나, MRP와 설계부문의 통합은 저조한 형편이다.

일본의 경우는 공작기계회사를 중심으로 하여 NC공작기계에서 머시닝센타로, 머시닝센타에서 FMC와 FMS로, FMS에서 CIM으로 一式의 제조현장을 점진적으로 통합 확대하여 나아가는 Bottom-Up 방식으로 CIM을 추진하는 경향이 있다. 따라서, 유럽이나 미국에 비하여 로보트와 FMS의 보급이 상대적으로 높은 편이고, DNC, FMS와 같은 기술들이 turn-key 시스템으로 취급된다. 물류자동화와 line 자동화 등 소위 FA의 확산에 비하여 생산통제와 OA 기술수준은 상대적으로 뒤떨어져 있는 형편

	product design	production control	production	office
state of the art	CAD used for design, drawings and archieving central NC-programming CAPP only in certain branches (aircrafts, cars)	MRP, MRPII no integration	Flexible manufacturing lines low requirements for flexibility low qualification of staff quality problems	high standard of office automation cost accounting finance
integrative approach	Group technology as planning aid : - product design - operation planning - set up of manufacturing cells pilot installations : CAD/CAPP/CNC in sheet metall treatment and car industry			
planning method	set up of comittees and pilot sites seldom long range planning			

<도3> State of the Art in USA (Source : IPK, Belin)

이다. <도4 참조>

유럽의 CIM접근방식은 Top-Down과 Bottom-Up의 복합형식으로, CIM시스템의 실사용자별로 자기에게 가장 적합한 Architecture를 Top-Down으로 설계하여 세상에 나와 있는 요소제품들을 채택함과 동시에 미진한 기술을 개발하여 접목시키는 형태를 띄고 있다. <도5>에서 볼 수 있듯이 공정관리(Shop Floor Control)가 설계부문과 제조부문을 잇는 핵심 요소의 역할을 하고 있고, 극도의 다품종 소량생산체제에 적응하는 생산 시스템 구현에 많은 노력을 쏟고 있다.

제2절 설계기술

여기서 設計 技術이라 함은 Design, Analysis, Process Planning을 표괄하는 넓은 의미의 설계를 稱하고자 한다. 설계기술에 있어서 최근의 세계적인 주요 Issue로는 다음과 같은 것들일 것이다.

- Concurrent Engineering
- Product Model
- Unified Modeler
- Feature-based Modeling
- Integrated Design Environment
- Intelligent CAD
- Exchange of Geometric Data
- Domain-specific Design System
- Design Theory/Methodology

	product design	production control	production	office
state of the art	2D-CAD only for drawing small range of systems NC-programming often in the shop floor small variety of products	MRP small range of systems Just in Time - possible because of planned over capacity No integration	FMS large series Automation of material flow High level of staff qualification	inefficient organisation low level of automation
integrative approach	CAPP-DNC-FMS as turn-key-systems Factory Automation(FA) but no CIM Islands of automation for manufacturing, assembly and handling			
planning method	Bottom up, principal of consensus, wish of harmony			

<도4> State of the art in Japan (Source : IPK, Belin)

	product design	production control	production	office
state of the art	2D-and 3D-CAD drawing great range of systems NC-programming central or decentral high variety of products	great range of MRP systems planning according to capacity load	NC, CNC, DNC FMS, FMC highly flexible high level of qualification	increasing office automation
integrative approach	CIM for small and medium size batch production of complex products with high variety SFC as Important mean of Integration			
planning method	Bottom up in Top Down frame			

<도5> Trends in Germany (Source : IPK, Berlin)

일반적으로, 설계에서는 제품의 기능적인 구조와 특성을 결정하고 해석에 의해 適否를 평가한 후 사양을 상세하게 記述하여 제조에 필요한 공정계획을 생성하는 일들이 이루어진다. 한 제품의 최종가격 중 설계 cost가 차지하는 비중은 15% 내외 밖에 안되지만, 設計 行爲가 끝나면 제품의 최종가격은 이미 80% 이상 결정되어 버린다는 데에 설계의 중요성이 있는 것이다. 따라서 設計時에 이미 나중에 제조에 미칠 영향이 충분히 고려되어 설계에 반영되어야 한다. CAD의 時代인 오늘날, 이와 같은 Concurrent Engineering을 구현할 수 있는 computer tool을 개발하는 것이 매우 큰 연구과제의 하나이며, 이것은 Wire-frame Modeler와 Surface Modeler, Solid Modeler를 一體化 시키고자 하는 Unified Modeler 개발 노력, 그리고 Conceptual Design과 Synthesis, Analysis 및 Process Planning을 동일 환경에서 수행하고자 하는 Integrated Design Environment 구현 노력들과도 脈을 같이 하고 있다.

異種 CAD 시스템 間의 정보교환을 가능케 하고자 하는 노력은 IGES(International Geometric Exchange Standard)에서 PDES(Product Data Exchange Standard)로 이전되고 있다. 미국의 PDES와 EC의 CAD*I 는 ISO의 STEP으로 수렴되는데 여기서는 vendor independent 한 neutral interface를 개발하고 있다. CAD*I를 이용한 Solid Modeler 間의 데이타교환 테스트에 의하면, CSG 구조에서 CSG, B-rep, polyhedron 구조로의 변환은 모두 가능하고, B-rep 또는 polyhedron 데이타구조에서 CSG로의 변환은 불가능하였으며, B-rep이나 polyhedron구조끼리의 변환은 가능하였다. (상용 Solid Modeler Bravo3, Euclid, ICEM, Proren, Romulus, Catia, Geomod, Technovision 등이 사용됨). ESPRIT 프로그램은 CAD*I 以外에 CADEX, NIRO, NEUTRABAS 등의 프로젝트도 추진하고 있다. CADEX는 CAD*I activity를 FEM 분야에의 접속까지 확장하여 그 실용성을 시험하고자 하는 것이며, NIRO는 로보트 프로그래밍에 필요한 Kinematic information을 STEP activity에 포함시키는 연구이다. NEUTRABAS는 STEP의 내용을 造船 분야에 적용.확장하는 프로젝트이다.

ESPRIT의 IPDES 프로젝트에서는 기존 Modeler와 통합될 수 있는 Knowledge-based Product Modeller를 개발하는 데, AI를 이용하여 공정계획을 자동으로 생성할 수 있게 하기 위하여 Form Feature를 도입하고 있다. 비슷한 내용으로, 박판(Sheet Metal) 및 배의 프로펠라와 같은 복잡한 형상의 설계에 적용할 수 있도록 하는 연구가 IMPPACT라는 프로젝트에서 진행되고 있다. 以上 一聯의 연구들이 이종 CAD/CAM시스템들 간에 표준 인터페이스와 데이타 구조를 이용하고 設計에 Form Feature를 이용함으로써, 그리고 Process Planning으로부터 Design으로의 feedback mechanism을 구현함으로써 통합(Integration)을 실현하려는 노력으로 모아지고 있다.

제3절 제조 기술

가공, 조립 및 검사를 포함하는 製造技術에서는 知能化와 시스템化(統合化) 추세가 두드러진다고 할 수 있다. 이러한 기술동향은

- FMC, FMS
- shop floor에서의 Vision 기술 活用
- fault-tolerant 제어 技法

- Robot Calibration
- In-process 측정검사
- wireless AGV
- self-guided control of robot
- CMM-CAD 통합
- decentralized postprocessing
- hologram 이용 기술
- 지능형 기계가공시스템
- programmable device 제어의 Open Modular Architecture

등의 최근 세계적인 연구개발 테마에서 잘 나타난다.

유럽의 ESPRIT/MOSAIC 프로젝트에서는 로보트, AGV, 기타 작동기계들의 제어시스템을 위한 Open Modular Architecture를 개발하여 各 장비와 적용분야에 적합한 제어기능의 범위를 쉽게 짜 맞출 수 있는 하드웨어 및 소프트웨어를 실용화하고자 하는 연구가 수행되고 있다.

Intelligent CAM 시스템(일본 北海道大學), Automated NC(美 Cimplex), 또는 차세대기계가공시스템(日 미쯔비시전기) 등의 idea는, CAD의 Product Model에서 출발하여 tool, fixture, material 등의 DB를 이용하여 set-up계획, fixture 계획, 공구선정, 가공순서계획, 가공변수의 결정, 공구경로 생성, scheduling, 원가 및 표준시간 산정 등을 自動으로 수행할 수 있는 시스템을 개발하는 것이다.

microprocessor를 내장한 공작기계 Controller의 계산능력이 急增함에 따라 기존의 postprocessing 기능을 NC Controller 쪽으로 이전하려는 경향이 Decentralized postprocessing 이라는 개념으로 선보이고 있다. 지금까지

는 프로그래밍시스템에서 postprocessing까지 마친 NC Code를 공작기계에 보내었지만, 여기서는 표준 CL data 까지만을 NC Programming에서 완료하고 공작기계에 dedicated된 NC Controller에서 postprocessing과 동시에 작업에 들어가는 방식을 취한다. 이 경우 postprocessing 기능을 포함하는 NC Controller의 값이 비싸지는 단점이 있지만, 한번 생성된 CL file은 어떤 기계에서도 reprogramming 없이 활용될 수 있어 portability 향상에 의한 생산성 제고가 기대된다는 것이다.

FMS는 高價의 설비이기 때문에 가동률을 최대로 유지할 수 있어야 한다. 히다찌정공의 Holonical FMC와 같이 scheduling 등 고도의 운용기술 없이 FMC를 하나의 기계 Unit 처럼 최대로 가동할 수 있게 하는 개념이 나오기도 하지만, 高度의 flexibility를 유지하기 위하여는 운용 소프트웨어의 지능화로서 해결하려는 노력이 지배적인 경향이라 할 수 있다. 즉, FMS의 제한된 규모에 대한 DB를 기초로 하여 Knowledge Base를 加味한 지능형 scheduling과 tool management, dynamic process planning 기능들을 실현하고 real-time simulation 기능까지 합하여 진 운용시스템을 개발하는 연구가 各國에서 시도되고 있다.

shop floor의 各 process에 있어서 Vision system의 활용, Calibration 기술의 접목, fault-tolerant 및 fuzzy control 기술의 적용, 각종 programmable device의 自律制御 기능 장착 등의 技術들이 제조기술의 중요한 추세이며, 아직까지는 등한시되어 왔던 3차원 측정기술과 설계시스템을 Interfacing 하는 연구(CAD-CMM)통합도 많이 이루어지고 있다.

제4절 생산통제 기술

Production Control의 개념은 미국의 "Production Planning and Inventory Control"에서 시작된다. 資材의 수배기간을 cover하는 범위에서 일정 periode의 需要를 기초로 제품의 생산량을 확정하고 필요한 구성부품과 자재의 소요량을 산출하던 初期의 發注点 방식(Order Point System)에 의한 생산관리는, 제조품목을 독립수요품목과 종속수요품목으로 나누어 後者에 대해서만 자재소요량을 계획하는 MRP-I(Matcrial Requirements Planning) 시스템으로 발전한다. 여기에는 BOM(Bill of Materials)이 가장 중요한 역할을 하는 정보인데, BOM은 設計에서 만들어지므로 CAD와 MRP의 Interfacing이 중요한 Issue의 하나가 된다. MRP-I에서 취하고 있던 Infinitc scheduling 방식이 실현 불가능한 제조오더를 생성하여 공정관리의 어려움을 가중시키기 때문에, 실제 available한 설비의 capacity와 actual load를 고려하는 finite scheduling과 예기치 않은 상황발생에 의한 편차를 보정해주는 dynamic scheduling 기법이 개발되어 MRP-II(Manufacturing Resources Planning)시스템이 生産管理의 主流를 형성하게 된다.

생산관리/통제 분야에 있어서 최근의 관심은 보다 User-friendly한 Software System을 개발하는 것 外에

- MRP-II의 큰 계획구간(planning horizon)과 실제 shop floor에서의 real-time 상황 間의 時間的인 gap을 연결하는 방법
- 지식공학(Knowledge Engineering)을 이용한 Dynamic Scheduling System
- Cell level 또는 multi-cell level의 계획 및 제어시스템
- Shop floor에서의 data collection

- 제품의 life-cycle tracking
- 분산형 실시간 제어시스템
- 작업자의 경험과 지식을 최대한으로 활용할 수 있는 Smart한 interactive 환경
- MRP-II 시스템과 JIT(Justin time) 관리 방식의 접목

등에 모아지고 있다. ESPRIT 등 유럽의 R & D 프로그램과 일본의 IMS사업도 이와 같은 테마를 집중적으로 취급하고 있다.

제5절 Networking 및 Communication 기술

미국의 대기업들이 단일 vendor의 Controller를 사용하여 제조셀(manufacturing cell)을 구성하는 것을 선호한다면, 유럽에서는 로보트나 단위기계 level에서도 multi-vendor solution을 추구하는 경향이 있는 것 같다. 前者의 경우는 다른 vendor들로부터의 다양한 제조셀들을 연결하는 것이 주관심사인 반면, 後者에 있어서는 factory backbone level의 file transfer 통신이 아니라 low level에서의 automation protocol이 요구된다. 이와 같은 배경이 각각의 통신기술개발 방향에 직간접적으로 영향을 주고 있음은 부인할 수 없을 것이다.

ISO의 OSI(Open Systems Interconnection) Reference Model은 다른 Vendor의 제품이나 Network를 상호연결하기 위한 標準을 제공하고 있다. OSI의 7-Layer Model은, 下部 level 에서의 Linking과 Routing에 관계되는 Physical Layer, Data link layer, Network protocol layer, 中間 level 에서의 end-to-end transmission을 위한 Transport layer, Session layer, Presentation protocol layer, 그리고 上部 level의 User가 사용하는 기능을 다루는 Application protocol layer로 구성되어 있다. 美 國務省이 제정한 TCP/IP(Transmission Control Protocol/Internet Protocol)나, GM 社의 MAP, Boeing 社의 TOP(Technical Office Protocol)들도 모두 7-Layer Model을 따르고 있다.

산업자동화 부문의 통신기술에 있어서 무엇보다도 MAP은 가장 큰 비중을 갖고 있다고 할 수 있다. User들 중에는 MAP의 필요성을 현실적으로 인식하여 초기 Version을 채택한 후 단계적으로 Update된 MAP을 구현하려는 부류와, 최종 Version이 나올 때까지 기다렸다가 완성된 MAP을 채택하려는 그룹이 있는데, MAP의 규격과 테스트에 관한 사항들이 완벽히 안정되기에는 좀더 많은 시일이 걸릴 것이다. 1988년 9월 Dallas에서 열린 MAP/TOP Users' Group 운영위원회에서는 MAP Version 3.0이 향후 6년간은 보장될 것이며 Version 4.0에 대한 시스템 테스트는 1994년 이전에는 실시되지 않을 것이라고 公示되었다. 제3장에서 서술될 GM의 MAP 5단계 계획이 끝나면, 하드웨어 및 소프트웨어 패키지를 maker로부터 구입하여 플러그만 꽂으면 그 device가 MAP에 실리도록 함으로써 CIM을 위한 tool을 제공할 날도 멀지 않았다고 장담하고 있다.

MAP Initiative는 일본에도 일련의 파급효과를 가져오고 있는데, 日本 通産省이 지원하는 FAIS(Factory Automation Interconnection System)프로젝트에 Mini-MAP이 반영되어 있다. 1988년부터 1991년까지 추진된 FAIS 프로젝트는 세계적인 MAP 추세에 대응하는 것이지만 제조부문에 한해서는 MAP과는 별도로 접속방식을 해결해 나아가려는 시도이기도 하며, 그 최종결과가 1992년 6월의 CIM Japan '92에서 demonstration될 예정이다. FAIS에서는 NC공작기계나 PLC, 로보트 등을 접속하는 방법으로서 MAP

이 가장 적당한가에 대한 의문을 가지고 공장 Networking의 새로운 해결책을 제시하고자 하는 것이다.

비슷한 脈絡에서 ISO/OSI spec을 수용하면서 MAP과 호환성을 가지고 - 前述한 바와 같이 - 유럽의 산업특성에 맞게 보완하려는 목표로, EC에서 ESPRIT 사업의 일환으로 추진하고 있는 것이 CNMA(Communication Network for Manufacturing Applications) 프로젝트이다. 여기에는 Siemens, Nixdorf 등의 Vendor와 British Aerospace 등의 대규모 User 들이 제휴하여 1단계사업(1985년부터 1988년까지)을 끝내고, 2단계 사업을 추진하고 있다. CNMA에서 수행되고 있는 주요 내용들은 아래와 같다.

- 유럽의 User's Requirements 도출
- MMS(Manufacturing Message Specification), FTAM(File Transfer Access & Management), Network Management 등의 profile 선정
- 선정된 profile의 구현 (heterogeneous한 환경下에서)
- 적합성 시험을 위한 tool 개발
- 적합성 테스트
- public demonstration
- 현장시험

ESPRIT의 또 다른 프로젝트 FICIM은, CIM을 위한 Fieldbus를 개발하여 CIM환경에 적용하고자 하는 것이다. MAP이나 CNMA의 高價문제를 해결하면서 CNMA network에 integration 될 수 있도록 추진되고 있는 이 프로젝트에서는, 여러 sensor와 actuator, 그리고 기타 low level device 들을 통합할 수 있는 solution을 추구하고 있다.

제3장 세계각국의 CIM기술개발 사례

제1절 개요

현대의 국제경쟁시장에서 生存의 수단으로 인식되고 있는 CIM기술의 확보를 위하여 세계각국은 다양한 전략을 펴고 있다. 유럽이 그 중소기업들의 CIM기술 확보를 위하여 범국가적으로 ESPRIT, BRITE, EUREKA 등의 대형프로그램을 지원하고 있는 반면, 일본은 굴지의 대기업을 중심으로 민간주도로 CIM기술을 개발하고 있고 최근에는 모든 기업과 전세계가 참여하는 IMS프로그램을 제창하고 있다. 미국도 대기업들이 자체 경쟁력 제고를 위한 대책의 일환으로 CIM기술개발을 추진하고 있는 상황이며, 국가가 지원하는 CIM기술개발 사례로는 NIST(National Institute of Standards and Technology)의 AMRF(Automated Manufacturing Research Facility)와 NSF(National Science Foundation)에서 지원하는 一聯의 ERC(Engineering Research Center)들을 들 수 있다. <표1>은 선진국의 CIM기술개발 프로젝트 例를 보여준다.

한편, 우리나라와 중국, 태국 등 개발도상국에서도 CIM기술의 중요성을 인식하여 기술개발이 시작되고 있는데, 아직은 주로 정부주도로 추진되고 있다. 중국과 태국 등은 전적으로 정부지원의 CIM기술개발 및 모델플랜트 구축이 수행되고 있으나, 우리나라의 경우는 최근 3 - 4년간의 CIM기술개발 국책연구사업을 기반으로 하여 産學硏 공동으로 추진하되 기업이 사업을 주관하는 방향으로 전환되고 있는 실정이다.

	프로젝트	추진주체	목표, 내용
미국	ERC's	NSF	CAD, CIM 등 기초기술 개발
	AMRF	NSF/NIST	유연성 생산시스템(FMS) 구현
유럽	ESPRIT	EC / ESPRIT 위원회	정보처리기술 (Information Technoloty) 고도화 - CIM요소기술 및 Pilot Plant 개발
	BRITE	EC / BRITE 위원회	유럽 중소기업의 기술력향상을 위한 생산기술 개발 · CAD/CAM 및 CIM기술 개발
	EUREKA	EC / EUREKA위원회	유럽기업의 경쟁력 제고를 위한 첨단기술 (High-Tech) 개발 - 미래형 서류없는 공장 모델 등
일본	IMS	통산성 / IRORA	차세대 고도생산시스템 개발 · 표준화 - 지능화

<표1> 선진국의 CIM기술개발 프로젝트 사례

제2절 유 럽

유럽공동체(EC)는 미국,일본에 대하여 산업기술면에서의 상대적 경쟁력 강화를 위하여 국제공동연구개발을 활발히 진행해 오고 있다. 30여가지에 이르는 범유럽적인 과학기술분야에서의 공동연구개발계획 중에서도, ESPRIT, EUREKA, BRITE/EURAM, RACE 등은 CIM과 관련된 프로그램으로 매우 중요한 사업들이다. 이들 프로그램의 공통점은 EC Commission과 기술개발의 실수요자가 공동 출자하여 산업체 Implementation을 목표로 하고 있고, 産.學 또는 産.硏이 함께 참여하지 않는 프로젝트는 지원되지 않는다는 것이다.

1. ESPRIT

ESPRIT(European Strategic Programme for Research and Development in Information Technology)프로젝트는 정보처리기술분야의 기반기술개발계획으로 2단계로 추진되고 있다. 제1단계(ESPRIT-I)에서는 1983년부터 1987년까지 7억5천만ECU(약 8억달러)의 연구비를 총 173개 세부과제에 지원하였고, 그 대상분야는

Advanced Microelectronics

Software Technology

Advanced Information Processing

Office System

CIM

의 5개분야로, <표2>는 ESPRIT-I의 CIM관련 주요 프로젝트를 보여준다. 제2단계(ESPRIT-II)계획은 1987년에서 1992년까지로 16억ECU(약 17억달러)의 연구비를

Microelectronics and Peripheral Technologies

Information Processing Systems

Information Technology Application Technologies

의 3대 중점추진분야에 투입하고 있다.

마이크로엘렉트로닉스 및 주변기술 분야는 고밀도 집적회로, 고속집적회로, 다기능집적회로와 그 주변기기 개발을 주 내용으로 하고 있고, 정보처리시스템 분야는 1990년대의 정보처리시스템 개발을 위한 시스템 설계기술, 지식공학, 고도시스템 아키텍처, 신호처리 기술개발이 주된 연구개발 내용이다. 정보처리 응용기술분야는 실제 산업환경에 정보처리기술을 적용하여 EC기업의 경쟁력제고를 도모하는 것으로, 그 세부분야 및 연구개발내용은 <표3>과 같다.

2. BRITE / EURAM

BRITE / EURAM(Basic Research in Industrial Technologies for Europe and on Advanced Materials)계획은

Advanced Materials Technologies

Design Methodology and Assurance for Products and Processes

Application of Manufacturing Technologies

Project No.	Project Title
688	OSA : A CIM architecture based on the Open Systems Approach
496	Projcet Design & Specification of Configurable Graphics Subsystem for CIM
322	Interfaces in CAD systems
477	Control Systems for Integrated Manufacturing - The CAM Solution
418	Open CAM System
534	Development of an Automated Flexible Assembly Cell
9	Optical Sensing Approaches for Adaptive Control of Arc Welding
118	General Purpose Sensory Controlled System for Parts Production
179	Integrated Electronic Sub-System for Plant Automation
338	Product Design for Automated Manufacture and Assembly
319	Data Transfer between CIM and Management Information systems
384	Integrated Information Processing for Design Planning & Control of Assembly
595	Application of CIM to Welded Fabrication
809	Advanced Control Systems for Flexible Manufacturing
932	Knowledge Based Real Time Supervision in CIM
1136	Distributed Automated System for Inspection and Quality Control

<표2> ESPRIT Phase I의 주요 연구과제

기술 분야	세부 연구개발 내용
Computer Integrated Manufacturing (CIM)	- Design and Analysis Systems Allowing Flexible Product Development - Factory Management, Planning and Production Control - Man-machine Interactions for Production Planning and Control Systems - Robotics Systems - Integration of Material Handling Systems (Including Robots) in the Production and Assembly Process - Computer Integrated Control in Process Industries - Architecture and Methods for Integration, Including the Development Methods and Tools for Installing, Operating, and Monitoring CIM System
Integrated Information Systems	- User Environment Analysis and Support - Systems Engineering Comtrising Systems Integration and Validation Tools - Generic Communication Technologies and Integrated Office Systems - Distributed Systems with Particular Emphasis on Integration of Knowledge Based Systems and Advanced Distributed Storage Systems
Information Technology Application Support Systems	- Workstation for Multiple Application Purposes - Storage and Processing Subsystems for Stand-alone and Distributed Systems - Local Network Systems and Related Basic Services - User Interfacing SubSystems - Subsystems Interfacing the Physical Environment

<표3> ESPRIT-II IT 응용기술분야의 주요 연구개발 내용

Technologies for Manufacturing Processes
의 4개분야에서 특히 중소기업의 경쟁력 강화를 지원하기 위한 프로그램으로, 실용화될 가능성이 높은 테마를 선정하여 EC지역내의 공동연구에 50%의 연구비를 지원하며 주관연구기관은 반드시 기업체가 하도록 되어있다. ESPRIT와 EUREKA계획이 전략성을 염두에 두고 있다면, BRITE프로그램은 실용성과 즉효성을 겨냥하고 있다는 점이 근본적인 차이라 할 수 있다. 의복, 제화, 선박, 육가공, 금형, 가구 등 산업분야별 CAD/CAM 및 CIM기술개발을 추진하는 이 프로그램의 대표적인 프로젝트는 <표4>와 같다.

3. EUREKA

EUREKA(European Research Coordination Agency)는 1985년 미테랑 프랑스 대통령이 제창한 첨단기술공동개발계획으로, 정보처리기술, 로보트 및 생산기술, 유전공학, 신소재, 환경, 통신, 에너지, 레이저 등의 분야에 대해 적용사례를 중심으로 연구개발을 지원하는 것이다. 총연구비는 65억ECU(약 70억달러)로 되어 있고, 지금까지 297개의 프로젝트가 수행되고 있다. CIM과 관련된 분야의 대표적인 과제는 <표5>와 같다.

4. RACE

RACE(R & D in Advanced Communication Technologies in Europe)계획은 EC의 통신산업의 경쟁력강화를 목적으로 1987년에 제안되어 1992년까지 5.5억ECU(약 6억달러)의 예산으로 진행되고 있다. RACE에서는 EC전역

Project No.	Project Title
1391	MODESTI(Mould Design and Manufacturing Optimization by Development, Standardization and Integration of CAD/CAM Procedures)
2406	Integration of CAD/CAM and Production Control for Sheet Metal Components Manufacturing
1320	CAD/CAM and Group Technology for the Furniture Industry
2478	Realization and Test of Prototype of a Basic Module of CIM in the Clothing Industry
2351	Development of a CAD/CAE System for Ship Design suitable for SMEs
2248	CIM System for the Meat Processing Industry
3447	TOPSYS - Tool Production System for Design and Manufacture of Models for High Quality and well fitting hoes in the Footwear Industry
3148	Production and Cost Oriented Mould Integrated Design Expert System (PROMISES)
3455/6	FEMOD- Application of Feature Based Modeling for Complex Product Design and Manufacture
3600	An Intelligent Knowledge Assisted Design Environment Incorporation Manufacturing and Production Information (IKADE)
2362	Modeling and Automatic Control of the Polishing Process (mac pop)
1381	Interactive Knowledge Based Shop Floor Control System in a Small Manufacturing Enterprise Environment

<표4> BRITE / EURAM 프로그램의 프로젝트 예시

Project No.	Project Title
64	Proposed development of a computerised engineering unit
68	FIELDBUS : Eurobot / Communication architecture Industrial LAN for real-time process and machine control
130	EUREKA JESSICA : CIM for constructional steelwork including expert systems
143	Automated cutting room for the leather industries
152	EUROPARI / SPIDER : Development of integrated flexible automated paperless design and production units - integrating all operations from design to inspection and product support
154	Factory of the future - Quality-oriented production in the future - Application of an expert system for CAD of work schedules and NC programs in the field of manufacturing of components - Expert systems for the design and control of flexible assembly systems - CIM assembly in precision engineering and electronics
256	FAMOS : Study of feasiblity for ECU production for automotive use
289	FAMOS / IDAP : Computer integrated design and assembly planning
21	PARADI : Development of flexible and interface systems
212	FAMOS/ARIA : Automated rapid integrated assembly
229	FAMOS/IDEM : Integrated design with engineering and automated manufacturing

<표5> EUREKA의 CIM분야 프로젝트 예

에 IBC(Integrated Broadband Communication)서비스를 겨냥하여

IBC의 개발 및 실용화 전략,

IBC 기술,

기능의 통합

등 3분야의 연구개발을 추진하고 있다.

제3절 미국

미국의 CIM기술개발은 주로 민간기업이 주도해 왔다고 할 수 있는 데, GM, Chrysler, McDonnell Aircraft, Ingersoll Milling, Deere, Westinghouse 등이 대표적인 例이다. 이들은 CIM기술의 도입을 통하여 2 -5배의 생산성향상, 총제작기간(Lead-time)의 30 - 60% 단축효과를 보고하고 있다.

GM의 경우, 1980年度에 공장자동화 투자의 50%정도가 機器들의 상호접속에 소요되어, 이들 異種기기간의 표준 통신규약에 대한 필요성이 제기되었다. 이를 위하여 개발된 것이 공장용 LAN의 표준규약(Protocol)인 MAP(Manufacturing Automation Protocol)으로, 1984년 Version 1.0이 발표된 이래 MUG(MAP Users' Group)이 결성되어 약 150여 회사가 참여하고 있고 현재 Version 3.0이 발표되어 있다. 이 표준 프로토콜에 의할 경우, 어떠한 Vendor의 장비라도 쉽게 접속이 가능하기 때문에, 미국뿐 아니라 유럽, 일본의 제조업체들도 MAP 호환 하드웨어 및 소프트웨어를 경쟁적으로 내어 놓고 있다. GM 社는 MAP의 실현에 있어 다음과 같이 5단계로 추진하고 있다.

Step 1 : 집중통신 (Centralized Communication) (1984년)

- IBM S/1을 프로토콜 변환장치로하는 스타형 모델

Step 2 : 프로그램 가동제어장치, 게이트웨이를 지닌 LAN(1984년)

- LAN에 의한 실현, OSI의 1,2층과 IEEE 802 표준과의 인터페이스 확립
- 일부 Programmable Device의 게이트웨이 개발

- 1984년에 NCC쇼에서 Demonstration

Step 3 : 응용 서비스 (Application Service) (1985년)

- 응용층 서비스기능의 실현
  (파일전송, 메세지 통신, 가상단말기능, 네트워크 관리 등)
- 광역 네트워크의 게이트웨이 접속
- 1985년 AUTOFACT로 Demonstration

Step 4 : 저가격 하드웨어 (1986년)

- MAP 하위층의 VLSI화에 의한 저가격화, 양산화
- IEEE 802.4의 베이스밴드방식과 브로드밴드방식간의 브릿지개발

Step 5 : 네트워크 유틸리티 (1988년)

- MAP 네트워크의 완성
- 프러그 호환성의 실현
- 1 ~7계층의 spec 완성

공공부문에서는 NSF(National Science Foundation)와 국방성이 기술개발을 주도하고 있는 데, NIST와 ERC들, 그리고 ICAM(미공군) 등은 그 대표적인 사례이다.

NIST는 8개의 Lab. 으로 구성되어 있는 국책연구기관으로, CIM관련 연구는 MEL(Manufacturing Engineering Lab.)이 담당하고 있다. NIST는 미국의 산업계를 위하여 2000년대에 가장 파급효과가 큰 기술 중의 하나로 CIM기술을 꼽고 있다. (13개의 Emerging Technology 中에서 Computer-integrated Manufacturing & Flexible Systems를 A급으로 분류 : Source "The Status of Emerging Technologies - An Economic/Technological Assessment to the Year 2000). AMRF(Automated Manufacturing Research Facility)는 미국 내의

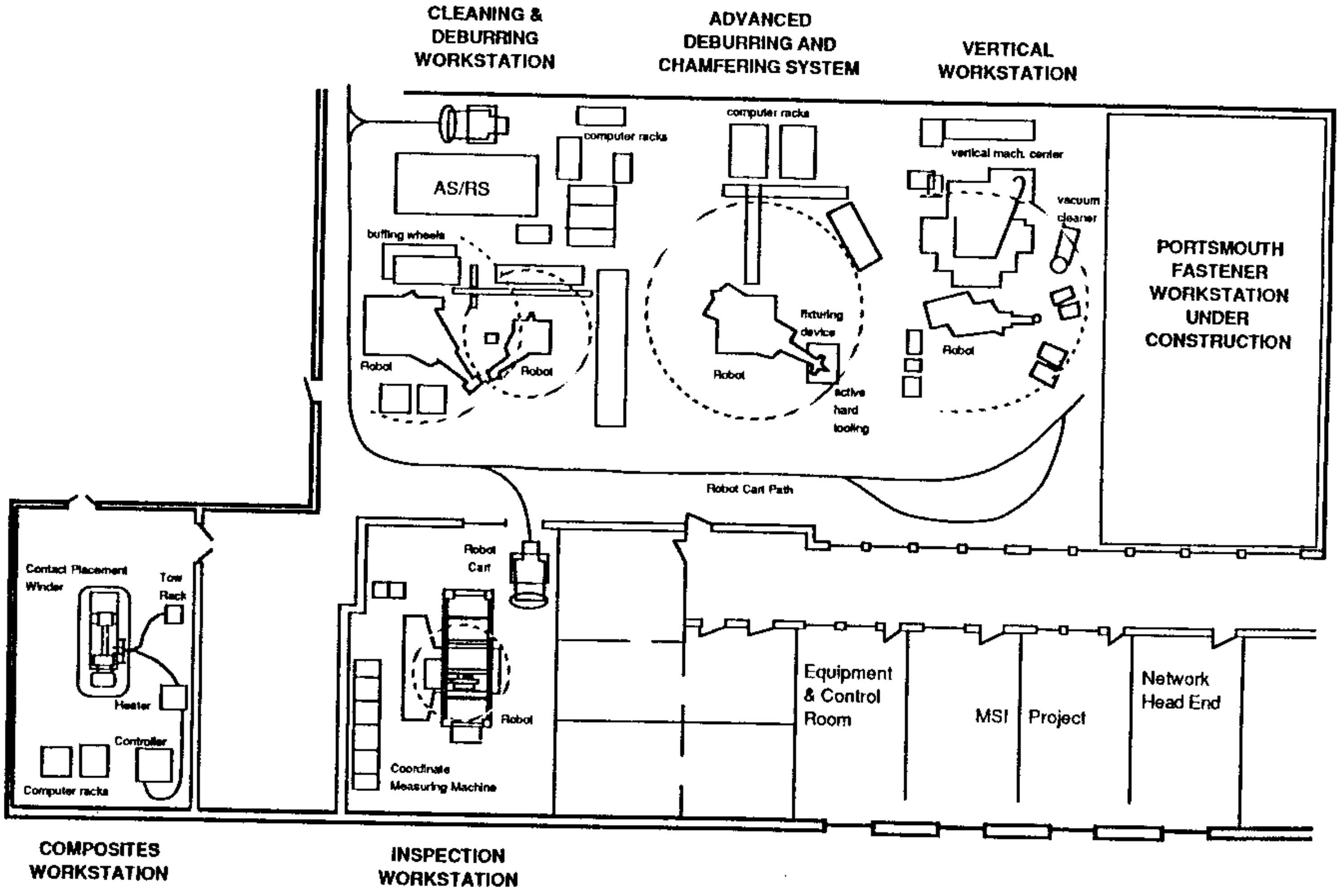
국가적인 평가척도가 될수 있는 정밀기계가공의 新技術을 시험하기 위합과, 장비들간의 표준 인터페이스를 개발하는 데 필요한 기술정보를 제공함으로써 美企業들의 생산시스템 근대화를 촉진시키기 위하여 설치된 것으로, STEP에 관한 연구의 testbed로도 사용될 계획이다. 이를 위하여 AMRF에서는 3대의 머시닝센타와 3차원 좌표측정기, 세척스테이션, 무인운반차, 로보트, 중간저장버퍼 등으로 구성되는 통합모델 플랜트를 구축하여 (도6참조),

- 로보트의 측정 제어기술 및 안전성
- 센서 응용 기술
- 공작기계의 정밀도 향상
- 공정설계
- 로보트와 각종 기기의 실시간 제어
- Cell Control
- 물류자동화

등의 연구를 수행하고 있다. AMRF의 연구결과는 ISO/TC 184(산업표준화시스템 및 통신기술)에서 미국이 주도하는 데에 크게 기여했다고 평가된다. NIST의 공장자동화시스템 Division은 AMRF외에도 많은 Facility를 보유하고 있는 데, 공학 표준개발 분야에서는

- Existing and emerging information technology standards
- Manufacturing data exchange standards
- Product data exchange standards
- Distributed data systems standards

등의 제정을 위한 연구가 진행되고 있다.


CLEANING & DEBURRING WORKSTATION
ADVANCED DEBURRING AND CHAMFERING SYSTEM
VERTICAL WORKSTATION
computer racks
AS/RS
buffing wheels
Robot
Robot
computer racks
fixturing device
Robot
active hard tooling
vertical mach. center
vacuum cleaner
Robot
PORTSMOUTH FASTENER WORKSTATION UNDER CONSTRUCTION
Robot Cart Path
Contact Placement Winder
Tow Rack
Heater
Controller
Computer racks
Robot Cart
Robot
Coordinate Measuring Machine
Equipment & Control Room
MSI
Project
Network Head End
COMPOSITES WORKSTATION
INSPECTION WORKSTATION

<도6> AMRF의 구성도

NSF가 지원하는 13개의 ERC(Engineering Research Center)중 5개 정도가 CIM기술분야에 지정되어 있는 데, 이들은 미국 대학의 생산관련 연구수행에 핵심적인 역할을 하고 있다. <표6>은 CIM관련 ERC의 명칭, 소재지 및 주요 연구내용을 보여준다.

Purdue 대학의 ERC-IMS(Intelligent Manufacturing Systems)는 최근 일본이 제창한 IMS 3극 프로그램의 명칭의 母體가 되는 terminology를 제공하고 있다. 여기서는, 다품종 소량생산체제를 위한 새로운 생산시스템에 관한 연구를 목표로 하여 高度設計生産 시스템, 시스템통합기술, 자율조정제어시스템 등에 관한 연구가 수행되고 있다. <도7>은 ERC-IMS의 장기적인 연구개발 계획을 나타낸다.

CMU의 EDRC는 과학적 설계(Science-based Design)를 위한 제반 기법들에 관한 연구와 그를 위한 Tool 개발을 주요 목표로 하여,

- 설계평가
- 형상 표현 및 Reasoning
- 설계 代案의 제시를 위한 지능형 시스템
- 최적화 기법
- Domain-specific Modeling Environment

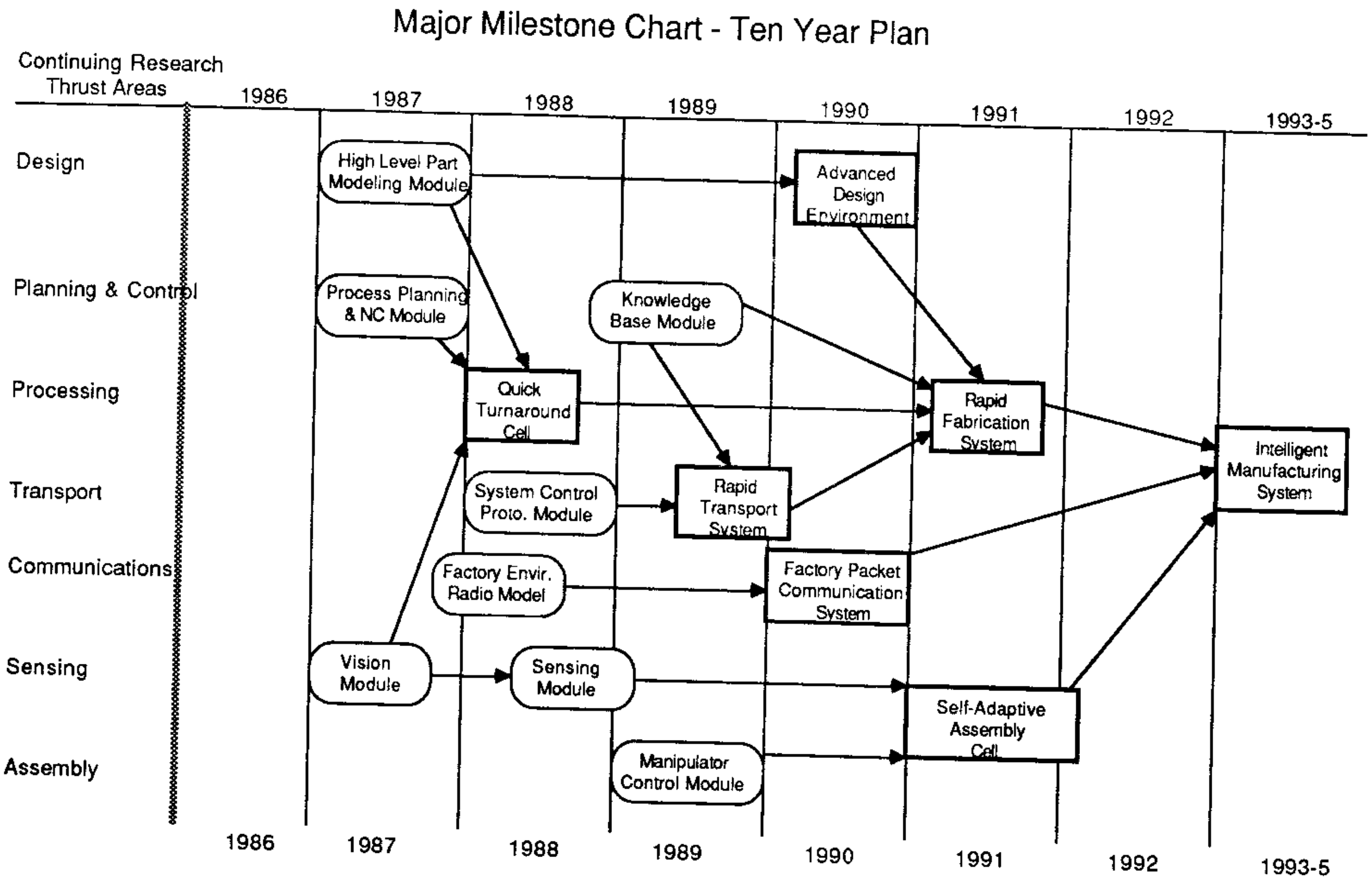
등의 연구개발을 추진하고 있다. 특이한 점 한가지는, 산업계 설계부문에 있어서의 하부구조(Infrastructure)개선을 위한 전략적 프로그램도 마련하고 있는 데, 그 장기계획이 <도8>에 圖示되어 있다.

Ohio주립대학의 ERC/NSM은 금형 등을 이용한 정형생산(Net Shape Manufacturing)의 합리화와 생산성향상에 관한 연구를 수행하고 있으며, 다음 6개 테마가 중점추진분야로 되어 있다.

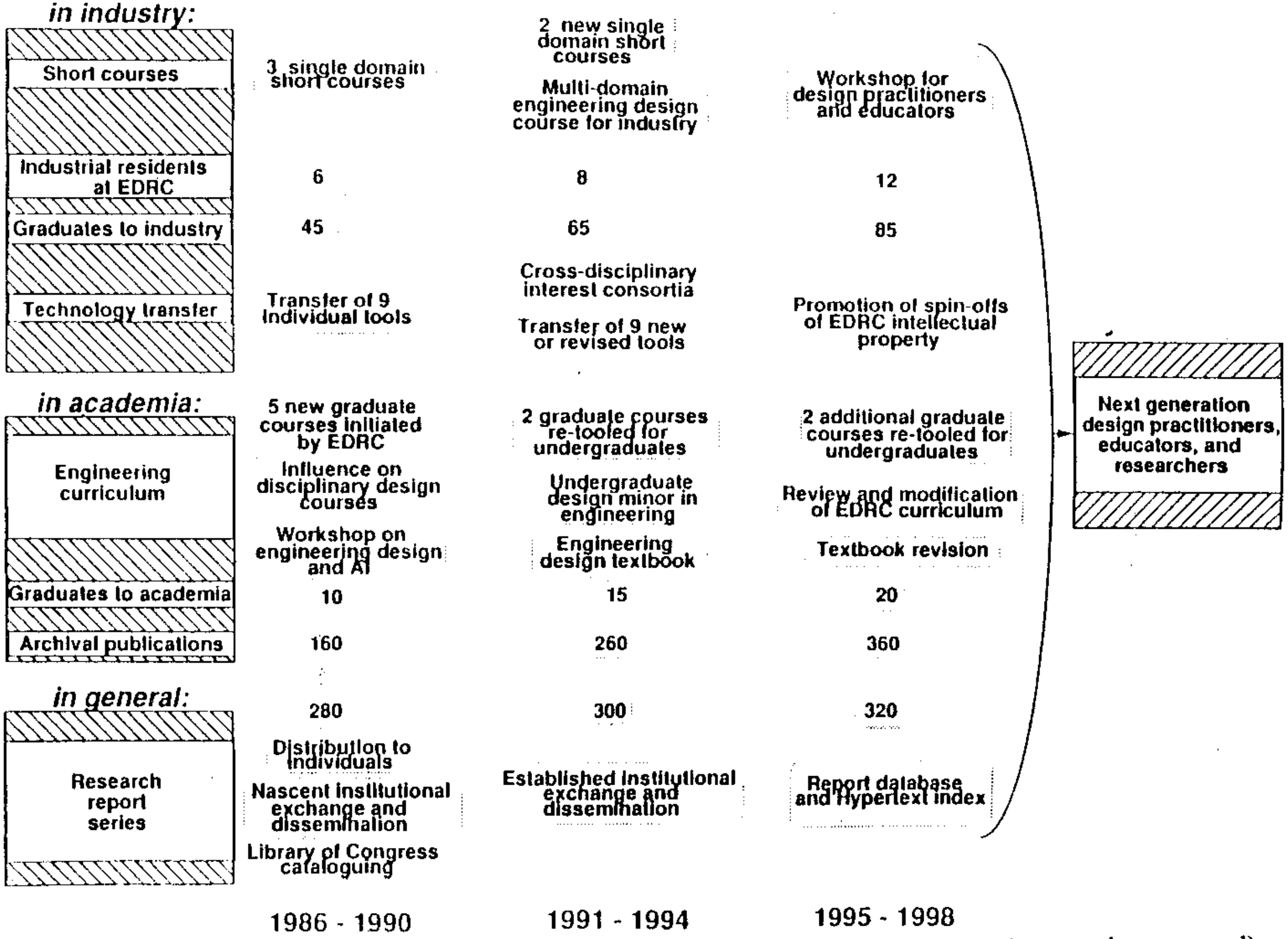
- Billet Forming

명 칭	주요연구내용	소재지
Engineering Research Center for Robotics Systems and Micro-electronics	Advanced Mechatronics Machine Perception and Intelligence Robotic Systems for Microelectronics	University of California at Santa Barbara
Systems Research Center	Intelligent Servomechanism 개발 Process Automation Manufacturing Automation and Integration	University of Maryland (*Harvard대학과 공동연구)
Engineering Research Center for Intelligent Manufacturing Systems	New generation of manufacturing systems for discrete products Advanced CAD/CAM Fully integrated, flexible, self-adaptive computer controlled system	Purdue University
Engineering Research Center for Net Shape Manufacturing	3차원 기하형태의 직접가공을 위한 Tool 개발 Expert System (for Manufacturability) Computer Modeling	Ohio State University
Engineering Design Research Center	Design Science의 기초연구 Computer-aided Design Tools Integrated Design Environment 개발 New Design Infrastructure	Carnegie Mellon University

<표6> CIM관련 ERC 현황 (Source : National Science Foundation)


Major Milestone Chart - Ten Year Plan
Continuing Research Thrust Areas
1986
1987
1988
1989
1990
1991
1992
1993-5
Design
Planning & Control
Processing
Transport
Communications
Sensing
Assembly
High Level Part Modeling Module
Advanced Design Environment
Process Planning & NC Module
Knowledge Base Module
Quick Turnaround Cell
Rapid Fabrication System
Intelligent Manufacturing System
System Control Proto. Module
Rapid Transport System
Factory Envir. Radio Model
Factory Packet Communication System
Vision Module
Sensing Module
Self-Adaptive Assembly Cell
Manipulator Control Module

<도7> Engineering Research Center for Intelligent Manufacturing Systems (Purdue 대학)의 연구개발 계획



	1986 - 1990	1991 - 1994	1995 - 1998
in industry:			
Short courses	3 single domain short courses	2 new single domain short courses Multi-domain engineering design course for industry	Workshop for design practitioners and educators
Industrial residents at EDRC	6	8	12
Graduates to industry	45	65	85
Technology transfer	Transfer of 9 individual tools	Cross-disciplinary interest consortia Transfer of 9 new or revised tools	Promotion of spin-offs of EDRC intellectual property
in academia:			
Engineering curriculum	5 new graduate courses initiated by EDRC Influence on disciplinary design courses Workshop on engineering design and AI	2 graduate courses re-tooled for undergraduates Undergraduate design minor in engineering Engineering design textbook	2 additional graduate courses re-tooled for undergraduates Review and modification of EDRC curriculum Textbook revision
Graduates to academia	10	15	20
Archival publications	160	260	360
in general:			
Research report series	280 Distribution to individuals Nascent institutional exchange and dissemination Library of Congress cataloguing	300 Established institutional exchange and dissemination	320 Report database and hypertext index

Next generation design practitioners, educators, and researchers

<도8> Strategic Plan for Institutionalizing a Design Infrasturcture (* All numbers are incremental)

Engineering Design Research Center (Carnegie Mellon 대학)

- Die and Mold Manufacturing
- Precision Casting
- Polymers Processing
- Sheet Metal Forming
- Systems Integration

제 4절 일 본

일본에서의 CIM에 대한 개념은 설계, 제조, 판매, 관리의 각 부문 정보를 컴퓨터에 의해서 통합화하고 생산의 효율화를 추구하는 것으로 인식되어 있으며 CIM의 발전 단계를 다음의 네 단계로 보고 있다.

제1단계 :자동기기 도입에 의한 단위 기계 자동화

제2단계 :복수기계의 접속에 의한 한 공정의 자동화

제3단계 :각 공정을 접속함으로서 제조현장 전체의 자동화

제4단계 :제조 현장과 사무관리 System의 통합

현재 일본 제조업의 대부분은 제1단계에서 제2단계로 진행되고 있는 과도기에 해당한다고 보인다. 절삭공정에서 보면 단일체 자동화의 완성품이라고 할 수 있는 Machining Center의 보급에 의해 거의 완성기에 들어 갔고 공정의 자동화로 향하는 제2단계로 진행되고 있다. 즉 수대의 NC기계를 Robot, AGV에 의해 접속, 공정을 자동화하는 FMC, FMS가 보급되고 있다. CIM이 3단계, 4단계로 진행되기 위해서는 MAP을 시작하는 통신 Protocol, 공장내 Network를 구축하는 것이 필요하다.

일본의 CIM기술개발은 전적으로 민간기업에 의해 이루어지고 있으며, 여기에 소개하고자 하는 IMS(Intelligent Manufacturing System)사업에서도 주관부처인 通産省과 그 산하추진기관인 IMS센타는 단지 Coordination 역할만을 담당할 뿐, 대부분의 연구비 出資와 연구개발은 참여기업들이 맡아 수행하는 것으로 되어 있다.

IMS(Intelligent Manufacturing System) 프로그램은 일본이 중심이 되어 미국 및 유럽공동체(EC)와 국제 공동연구로 21세기를 지향하는 새로운

고도생산시스템을 개발하는 10개년 연구사업으로, 일본에서는 日, 美, EC의 三極 프로그램이라 불리워 진다. 이를 위하여 일본은 6억불을 투자할 계획을 가지고 미국과 EC에 대해 각 각 2억불씩을 투자할 것을 요구하여 10억불 규모의 연구개발 프로그램으로 구상하고 있다. IMS란 본래 미국 과학재단이 지원하여 1986년에 시작된 Purdue 대학의 ERC 명칭을 그대로본따 그 개념과 범위를 확대시킨 것이다. IMS란 인간과 기계의 관계를 보다 중시한 21세기형 CIM으로서, 제조업에 있어서의 각각의 知的인 활동을 살리고 知能化된 기계와 인간의 융합을 도모하면서 수주에서부터 연구개발, 설계, 생산, 판매, 아프터서비스까지의 기업활동 전체를 flexible하게 통합. 운용하여 생산효율을 극대화하는 시스템을 개발하고자 하는 것이다.

IMS의 연구 영역은

기존 요소기술의 정비 및 체계화,

현재 및 차세대 생산기술의 표준화 추진,

차세대 고도 생산기술 개발

등 3분야인 데, 표준화와 순수 및 기초 연구에 중점을 두고 있다(도9 참조).

IMS의 목표와 관련하여는, 연구 개발과 생산 및 영업, Logistics, 외주선 등 제조업을 구성하는 모든 부문에 지능형 또는 자율 고성능 시스템을 적용하여 인간과 기계가 조화된 공장을 구성해 보자는 개념적 목표 이상은 아무것도 정의되어 있지 않다. (도10 참조) IMS의 내용은 Feasibility Study가 수행된 후에 확정될 예정이기 때문에 아직 구체적인 것은 나와 있지 않지만, 1990년 5월의 계획안에 의하면 다음 다섯 분야

가 중점 추진 연구 분야로 선정되어 있다.

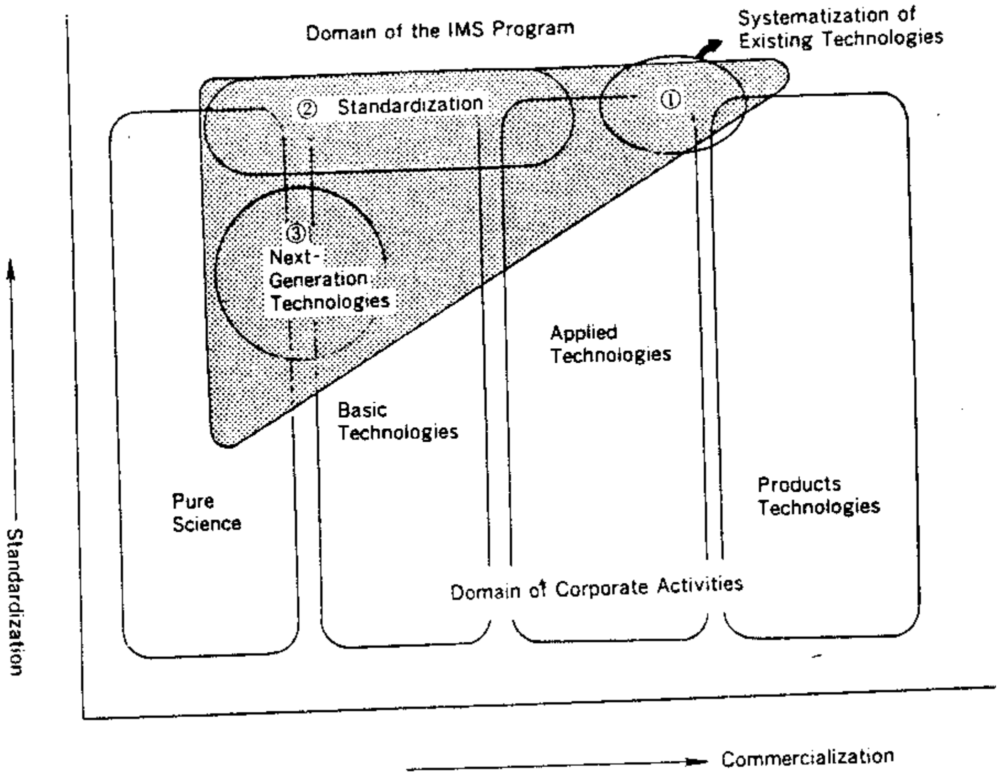
시스템 구성기기 가공 기술

시스템 설계 및 구축 기법

경영정보 통합화 기술

사회 환경 적응화 기술

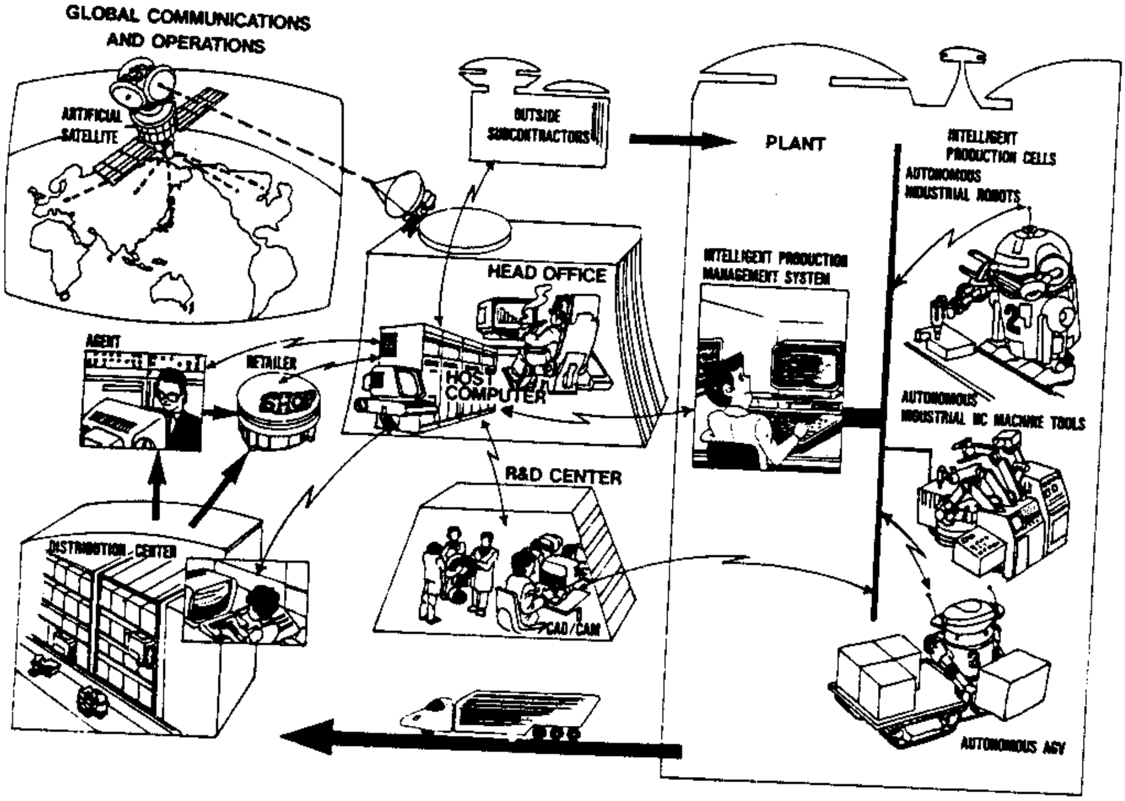
각 산업에의 응용 기술


Domain of the IMS Program
Systematization of Existing Technologies
② Standardization
①
③ Next-Generation Technologies
Applied Technologies
Basic Technologies
Pure Science
Products Technologies
Domain of Corporate Activities
Standardization
Commercialization

<도9> IMS프로그램의 연구 영역(Domain of Research)

IMS 프로그램은 동경대학 요시까와(吉川) 교수의 Techno-Glabalism 철학을 배경으로 하고 있다. 그에 의하면, 세계 경제에 있어서 국가간 상호 의존의 심화와 다국간 협력 증진의 결과로 국가간의 기술 수준은 평

준화되어 가고 있는데, 이 기술 평준화 과정은 자유시장경제에서의 기술 보호주의, 기술적인 체계확립의 미흡, 급변하는 환경에서의 표준규격의 불일치, 그리고 국가간 언어 및 문화 장벽 등에 의해 지연 내지 방해되어 왔다. 견실한 경제력과 기술력으로 세계 경제에 막대한 영향력을 행사할 수 있게 된 일본은 이젠 국제적 기술 수준의 평준화를 위하여 능동적으로 기여해야 할 의무가 있다. 그에 필수적으로 요구되는 기술의 체계화 및 표준화를 위하여 기술 선진국인 미국, EC와 함께 공동 노력을 하자는 것이 IMS 3극 프로그램 제안의 배경이라고 주장되고 있다.


GLOBAL COMMUNICATIONS AND OPERATIONS
ARTIFICIAL SATELLITE
OUTSIDE SUBCONTRACTORS
PLANT
INTELLIGENT PRODUCTION CELLS
AUTONOMOUS INDUSTRIAL ROBOTS
HEAD OFFICE
INTELLIGENT PRODUCTION MANAGEMENT SYSTEM
AGENT
RETAILER
SHOP
HOST COMPUTER
AUTONOMOUS INDUSTRIAL NC MACHINE TOOLS
R&D CENTER
DISTRIBUTION CENTER
CAD/CAM
AUTONOMOUS AGV

<도10> IMS에 의한 미래공장 개념도

이에 대하여 미국과 유럽은 일본의 제안을 선진 기술을 공개적으로 이전받기 위한 제스처로 해석하여 냉담한 반응을 보였으나, 점차 선

의적인 해석과 그 필요성에 대한 공감이 확산되면서 3극 프로그램 협의를 위한 실무회의가 개최되기에 이르렀다. 1991년 11월 동경에서 개최된 회의에서는 IMS 추진에 대한 의견수렴이 이루어 졌고, 일반 원칙 및 실시체제에 대한 검토와 Feasibility Study 수행에 대한 합의가 있었다.

일본은 IMS 추진을 위해 국제 로보트 - FA 기술센터 (IROFA) 내에 IMS 센타를 설치하여 IMS 프로그램에의 참여 기업을 모집하고 연구계획서를 받는 등 활동을 개시하였다. 1991년 7월 현재, 67개의 Core Member 기업과 20개의 Supporting Member 기업이 등록되어 있고, 118개의 연구계획서가 접수되었다 (표9 참조). 국제적인 차원의 Feasibility Study 수행이 지연되고 있는 중에, 일본은 IMS센타 주관으로 Task Force Team을 구성하여 이미 1차 Feasibility 조사를 1991년 7월에 끝내고 다음 단계로 나아갈 준비를 하고 있는 상황이다. Feasibility Study는 IMS센타가 주관하고 도시바, 미쯔비시전기, 히다찌제작소, 동양엔지니어링, 후지전기들이 세부분야의 총괄을 맡아

知能化 機器

自律制御

시스템 設計

시스템 構築

시스템 統合

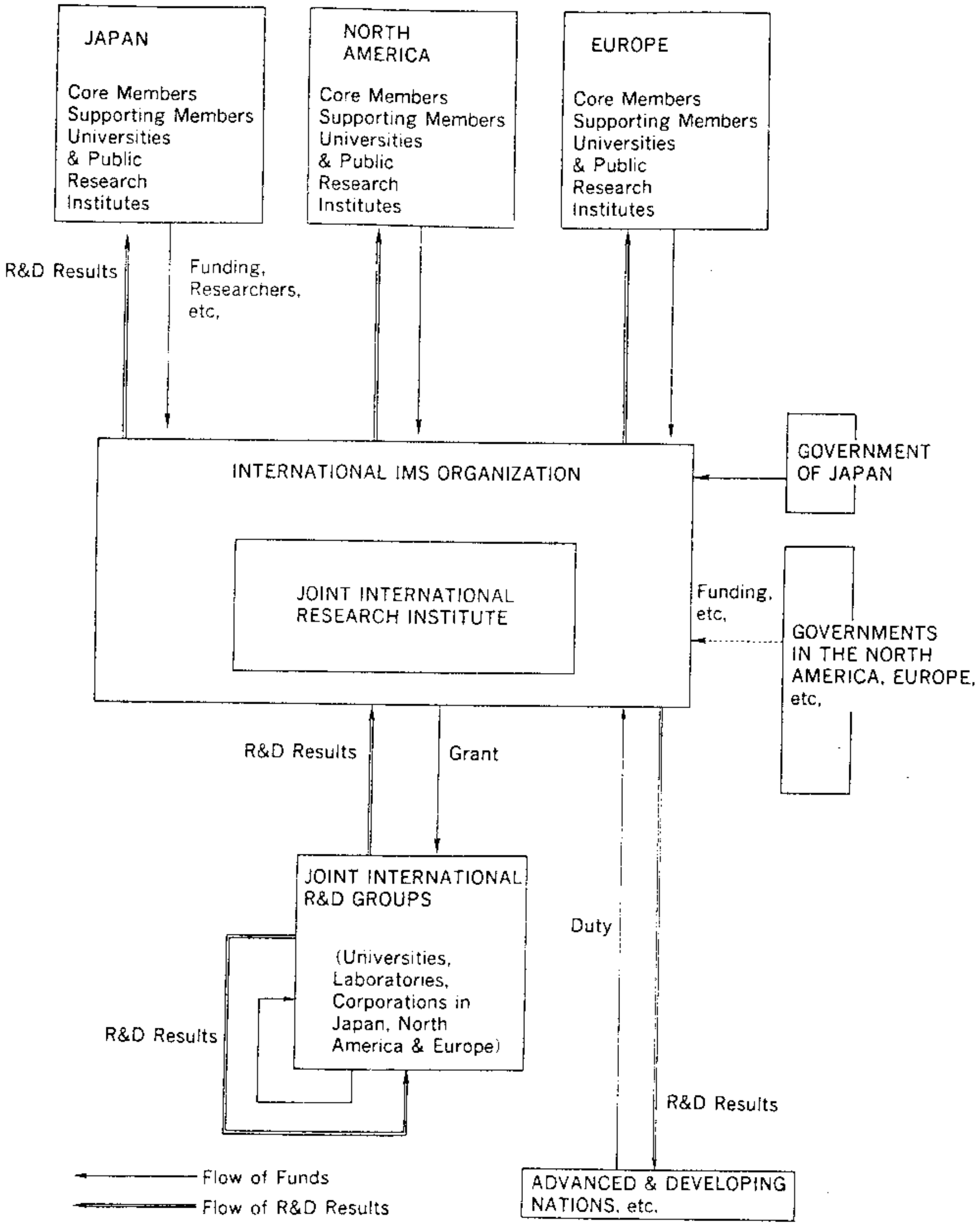
의 5개 부문에 걸쳐 수행되었다.

분 야	연 구 과 제
생산시스템 개발기술	자율분산형 생산시스템 지능형 CAD 시스템 자율 분산형 생산일정계획 시스템
정보처리 및 정보교환기술	생산시스템의 실시간 (real-time) 분산형 제어기술 Fuzzy logic과 Neuro logic을 이용한 생산통제기술 생산설비의 remote control 기술
생산설비 및 제조기술	생산시스템의 Fault-tolerant 기술 자율 고기능 로보트 AI를 이용한 자동검사기술 클린룸에서의 AGV 기술
신소재의 응용분야	차세대 센서 홀로그램 (Hologram) 기술 적외선 카메라 장치
생산에 있어서의 Human Factors	고도자동화 공장에서의 인간의 역할 멀티미디어 이용시 Human interface 생산환경 최적화 On-line Consultant Technology

<표9> IMS 프로그램의 연구개발과제 예시

以上 Techno-Globalism의 원칙론을 긍정적으로 조명하였으나, 실제로 일본이 기술의 국제 평준화에 기여하고자 하는 의도를 가지고 있을까 하는 부정적 견해는 여러 사례에 의해 뒷받침된다. 즉, 일본이 IMS 프로그램의 창시국으로서 한국 등 개발도상국들의 참여를 배제하고 있는 점, 사업의 본격화를 위해 필요한 Feasibility Study가 지연되자 단독으로 조사사업을 수행하고 일본 국내 기업으로만 구성된 회원제도를 도입한 점 등을 고려할 때, 일본이 세계 발전과 복지에 기여하겠다는 선의보다는 향 후 21세기에도 기술 우위를 확보하여 세계 경제를 주도하겠다는 의도를 읽을 수 밖에 없는 것이다. 요시까와 교수는 세계 경제의 상황이

현재의 Product Globalism/Knowledge Nationalism의 환경에서 미래에는 Product Nationalism/Knowledge Globalism 환경으로 변화될 것을 예견하고 있다. 즉, 각 국 특유의 Know-how에 의한 제조기술로 제품을 생산하여 세계시장에서 Marketing하는 현재에는 제품기술의 특허가 상품가치를 갖지만, 미래에는 지역의 특성에 맞는 최고의 제품이 소비자와 가까운 곳에서 제조.소비되는 경제 구조가 예상되며 이 때에는 지식이 무역의 주가 되어 제조기술에 대한 지적소유권이 상품가치를 갖게 된다는 것이다. 이와 같은 Knowledge Globalization 환경에서 지식이 상품화되기 위한 전제 조건으로서 제조기술에 대한 표준화가 필요하며 그 주도권을 일본이 확보하고자 하는 것이 아닌가 하는 것이다.



<도11> International IMS Research Program Scheme

제5절 개발도상국

개발도상국에 있어서는 CIM기술을 선도한다기 보다는 향후 예상되는 CIM기술에 대한 國內수요를 자체적으로 공급하기 위한 기술확보의 측면이 더 중요하다고 볼 수 있다. 따라서, 이들에 있어서의 과제는 CIM기술의 발전동향을 조속히 파악하여 좇아가는 데에 있으며, 中國이나 우리나라를 포함하는 NIES 국가들의 사정은 모두 비슷하다 할 수 있다.

중국은 1987년부터 High-Tech 연구개발을 체계적으로 기획, 추진하고 있다. 국가과학기술위원회(SSTC : State Science and Technology Commission) 산하의 자동화기술 전문가위원회는 CIM 전문가 그룹과 로보트 전문가 그룹을 두어, 각각 대형 프로젝트를 수행시키고 있다(도12, 도13, 참조).

CIMS/ERC는 CIM모델플랜트를 구축하여, 연구개발의 testbed로 삼고 시스템통합기술을 개발하며 미래에 대비하고자 하는 것으로,

System Design Methodologies and Software Engineering,

System Simulation,

Network Research,

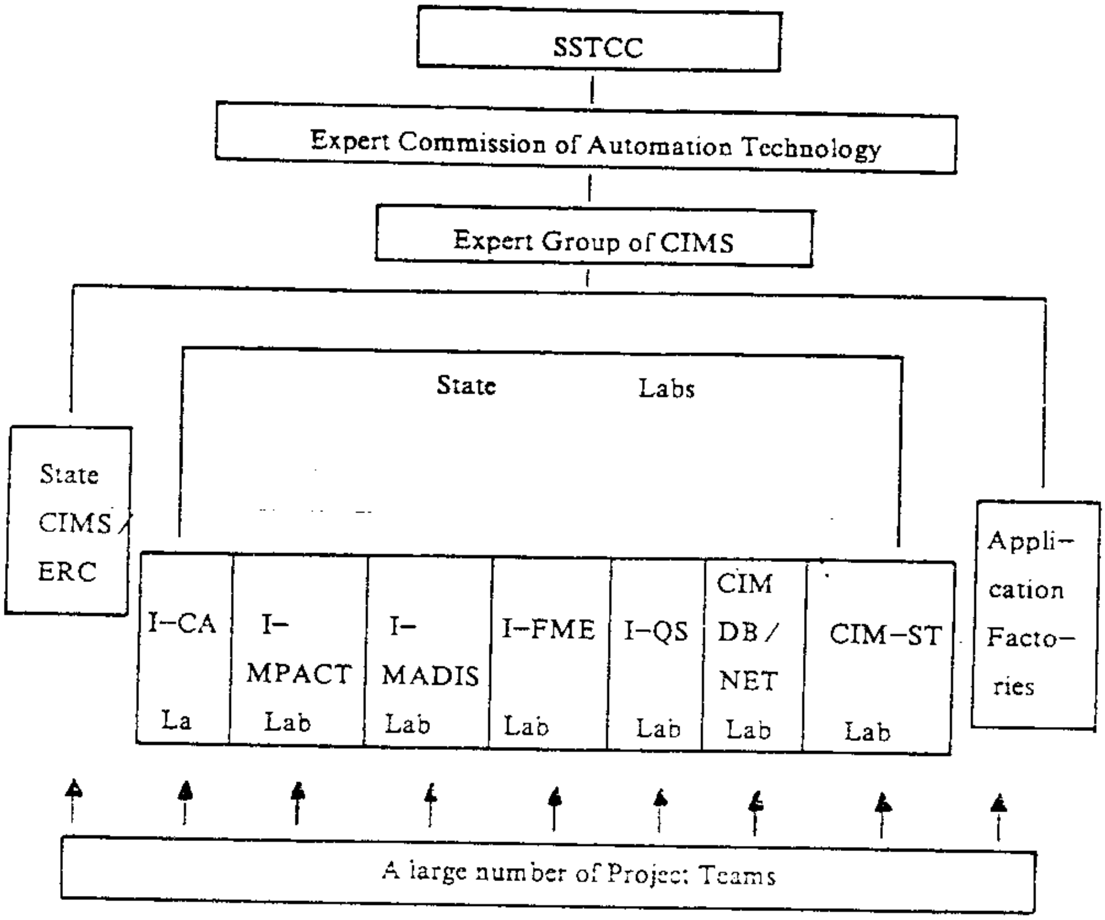
Database Research,

CAD/CAPP/CAM,

Hierarchical Control,

Manufacturing Systems Research

의 7개 부문으로 구성되어 있다. <도14>는 모델플랜트의 구성을 보여주는 데, 국내의 User들이 최적 요소기술을 선정할 수 있도록, 국내외의 기



Note:

CIMS / ERC: Engineering Research Centre of CIMS

I-CA: Information technology for / and CA technology

I-MPACT: Integration of Manufacturing Process Planning and CAD Technology

I-MADIS: Integrated Management Decision Information System

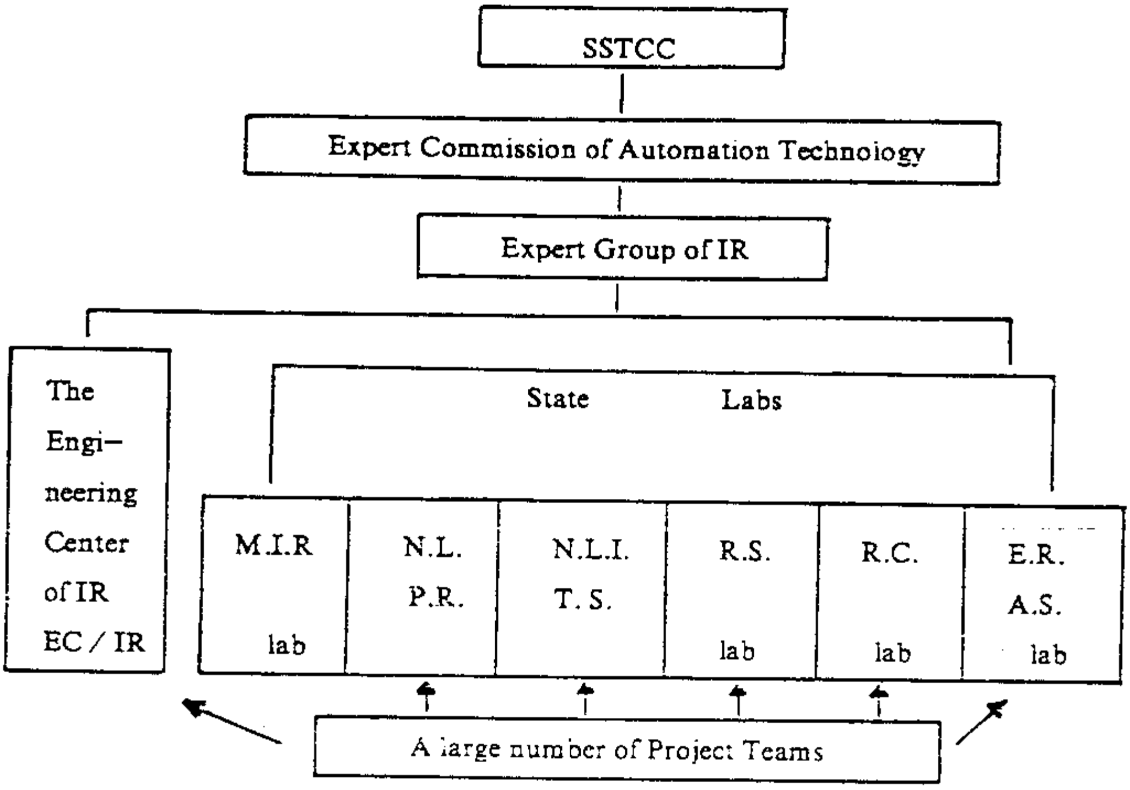
I-FME: Integrated Flexible Manufacturing Engineering

I-QS: Integrated Quality control System

CIM-DB / NET:Data Base and Network for CIMS

CIM-ST: System Technology for CIMS

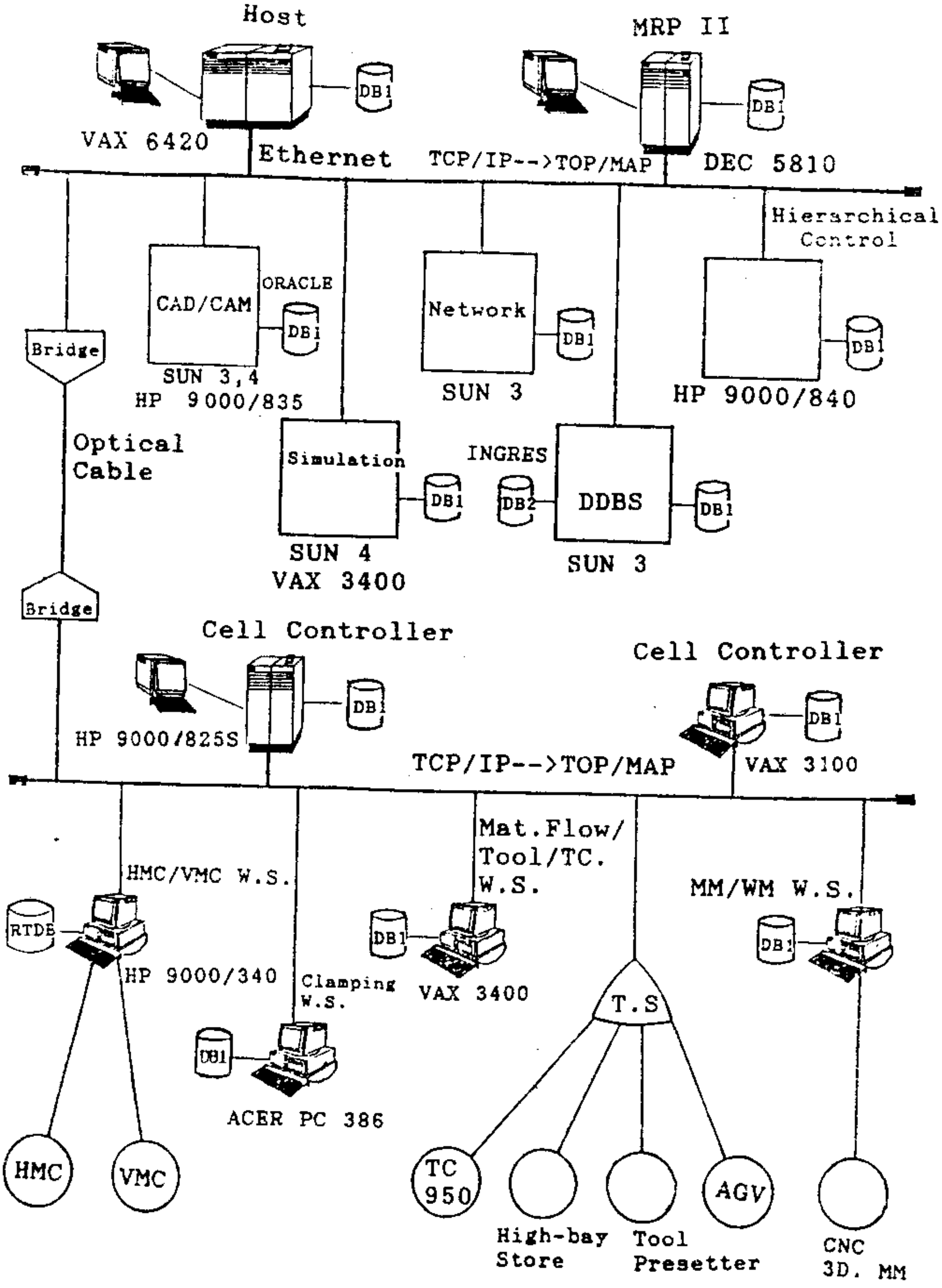
<도12> CIMS/ERC 추진체계


SSTCC
Expert Commission of Automation Technology
Expert Group of IR
The Engi-neering Center of IR EC / IR
State Labs
M.I.R lab
N.L. P.R.
N.L.I. T. S.
R.S. lab
R.C. lab
E.R. A.S. lab
A large number of Project Teams

Note:

E.C / I.R.–Engineering Center of Robot
M.I.R.L.–Mechanism of Intelligent Robot Laboratory
N.L.P.R.–National Laboratory of Pattern Recognition
N.L.I.T.S.–National Laboratory of Intelligent Technology and Systems
R.S.L.–Robot Sensor Laborotory
R.C.L.–Robot Control System Laborotary
E.R.A.S.–Experimental Robot Assembling System

<도13> ER/IR 추진체계



<도14> CIMS/ERC의 모델플랜트 구성

계와 컴퓨터, 소프트웨어들이 총동원되어 비교평가할 수 있는 시설을 갖추고 있다.

EC/IR은 고정밀 조립로보트와 해저작업 로보트, 그리고 이동로보트등 3종의 로보트 시제품을 개발하는 것을 목표로 하여 설립되어 아래 7개 부문의 연구개발을 수행하고 있다.

System Architecture of Robot,

Mechanism of Robot,

Robot Control,

Robot Vision,

Force/torque sensor and other sensors,

AI in Robot,

New structure material and special functional material, Energy source, communication, etc.

우리나라에서도 1988년부터 국책과제로 CIM기술개발사업이 체계적으로 시작되었다. "CIM기술에 의한 금형공장자동화"라는 副題를 갖는 이 사업은 종래의 금형기술에 CIM기술을 접목하여 금형생산의 납기를 획기적으로 단축하고 생산성을 향상시키기 위한 연구개발 프로젝트로, 금형의 설계부터 가공.검사까지의 전 과정을 일원화하고 최적화한 CIM 모델 플랜트를 구축하여, 현재 국내 사출금형 및 프레스금형공장의 생산성을 2배로 향상시키고 금형생산의 납기를 1/3 단축하는 것을 목표로 하고 있다. 이를 달성하기 위하여

I. CIM System Architecture 설계

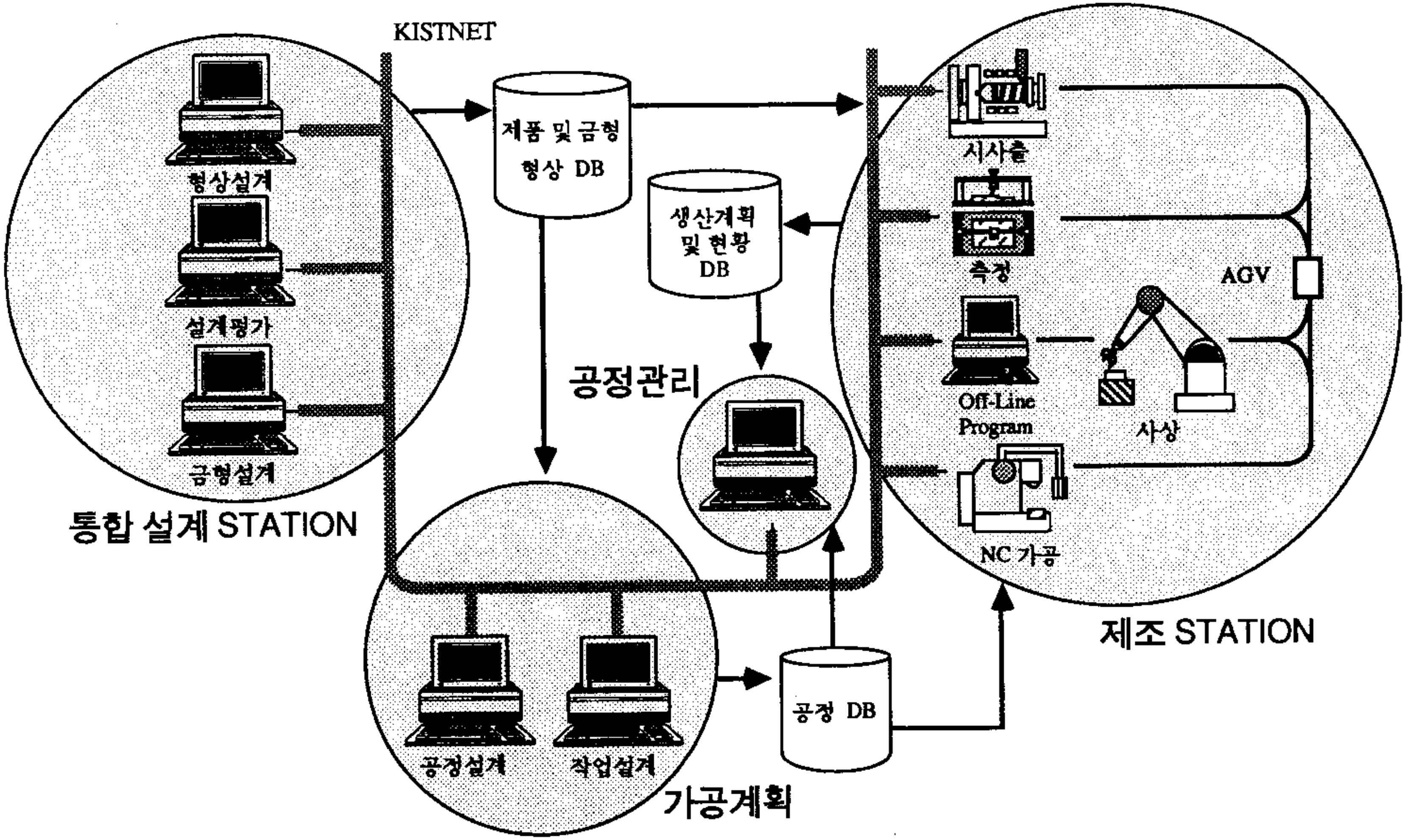
II CIM요소기술 및 통합기술 개발

- ◆ 설계.해석 및 평가기술
- ◆ 공정설계 및 NC기술
- ◆ 연마자동화 기술
- ◆ 측정.검사 자동화기술
- ◆ 정보시스템 및 생산통제기술
- ◆ CIM을 위한 표준 Network System

III. CIM모델 플랜트 구형

- ◆ 사출금형생산 모델플랜트
- ◆ 프레스금형생산 모델플랜트

등의 연구가 추진되고 있다. <도15>는 KIST에 설치된 금형생산 CIM 모델플랜트의 개념도를 보여준다.



<도15> 금형생산 CIM Model Plant

제4장 CIM분야의 국제표준

제1절 현황

CIM에서 추구하는 統合化 生産이 실현되려면, 생산시스템을 구성하는 모든 要素 - 설계시스템, 해석시스템, 가공시스템, 생산관리시스템, 품질관리시스템, 영업시스템, 그리고 모든 하드웨어 - 들이 서로 맞물려서 이들간에 정보교환이 이루어 질 수 있어야 한다. 수많은 CIM시스템의 User들과 요소제품의 Vendor들 사이에는 m : n 의 매우 복잡한 combination이 가능한 데, 이 때 발생할 수 있는 경우의 數는 가히 천문학적인 숫자이다. 따라서, 이 모든 경우에 compatibility를 보장하기 위해서는 표준 Interface에 관한 規約이 필요하게 되는데, 이를 産業標準(Industrical Standards)이라 한다.

CIM의 標準化를 추진하는 主體는 세계적으로 여럿이 있겠지만, 그 중에서도 다음의 Activity들이 가장 중요하다.

ISO/TC 184 - Industrial Automation Systems and Integration/Architecture and Communication Systems

ISO/IEC JTC1 - Information Technology

ESPRIT - AMICE (CIM-OSA)

CAM-I

MUG (MAP Users Group)

ISO는 국제표준기구(International Standardization Organization)로서 그 산하에

CIM을 관장하는 TC 184 外에도 全部門에 걸쳐 표준화를 주도하고 있으며, JTC1은 정보처리 분야에서 ISO와 IEC(International Electrotechnical Commission)가 Joint하여 구성된 Committee이다. MUG는 前述한 바와 같이 General Motors가 주관하는 공장용 표준 Networking의 사용자 그룹이고, CAM-I는 CAD/CAM/CIM 분야의 공통애로기술개발을 위해 결성된 미국의 산업체 Consortium인데 표준화연구도 수행하고 있다. AMICE는 European Computer Integrated Manufacturing Architecture를 제정하는 ESPRIT project의 하나로, CIM시스템의 광범위한 확산을 위한 Open Systems Architecture를 추구한 Activity이다.

이 外에도 DIN, AFNOR, ANSI, JIS 등 각국의 標準機構들이 자체적인 표준화를 추진하고 있으나, 이들의 표준도 점차 ISO의 국제표준에 맞추어 가고 있는 추세이다. CIM의 각 기술부문에 있어 현재 추진되고 있는 국제표준은 아래와 같다.

Graphics	PHIGS(Programmer Hierarchical Graphics System)	: ISO
	GKS-3D(Graphical Kernel System-3D)	: ISO
	CGI(Computer Graphics Interface)	: ISO
	CGM(Computer Graphics Metafile)	: ISO
Drawings/Geometry	IGES(Initial Graphics Exchange Specification)	: ANSI
	STEP(Standard d'Exchange et de Transfert)	: AFNOR
	VDAFS(VDA Surface Interface)	: DIN
Product Models	PDES(Product Data Exchange Specification)	: NIST
	STEP(Standard for the Exchange of Product Model Data)	: ISO

Protocols	MAP(Manufacturing Automation Protocol)	: ISO
	TOP(Technical and Office Protocol)	: ISO
	PROFI(Process Field Bus)	: DIN
Machine control	CLDATA(Cutter Location Data)	: ISO
	IRDATA(Industrial Robot Data)	: ISO
	APT(Automatically Programmed Tools)	: ISO

上述된 많은 표준화 Initiative들의 동향을 살펴보면, ISO를 國際標準化의 總本山으로 하여 ANSI, DIN, AFNOR, BS 등 각국의 표준화 Activity와 CAM-I, ESPRIT 등 연구프로그램의 표준화 노력 결과들이 ISO Document 수용되는 추세이다. 따라서, CIM의 표준화에 관하여 국제적인 동향을 추적하고자 하면 ISO/TC 184와 ISO/IEC JTCI의 Activity를 파악하는 것이 가장 중요하다.

제 2절 ISO/TC 184

19세기 말에서 20세기 초에 걸쳐, 서유럽에서 발생.완료된 근대 산업혁명은 대량생산과 국제분업을 낳게 되었으며, 이 새로운 생산방식은 대규모의 국제간 물자유통을 필요로 하였고, 이에 따라 필연적으로 '국제표준'의 제정이 중요해졌다. 이러한 필요성에 따라, 서유럽 국가들의 주도로 1926년에 설립되어 활동해 왔던 ISA(International Standardization Association)를 전신으로 하여, 1947년에 독립된 국제 기구인 국제 표준화 기구(International Standardization Organization)가 발족되었다.

ISO는 현재 각국을 대표하는 90개의 회원 단체로 구성되어 있으며, 전기 전자기기에 관한 표준을 제외한 전 산업에 걸친 표준을 제정

하고 표준화 활동을 지원하고 있다. 전기 전자기기에 관한 표준은 별도의 기구인 IEC(International Electrotechnical Commission)에서 담당하고 있다. ISO의 기술적인 작업은 ISO 가입국들의 자발적인 참여에 의해 이루어지며, 각 기술분야에 164개의 기술위원회(Technical Committee), 644개의 부위원회(Sub-Committee), 1,600여개의 작업반(Working Group)이 표준화 작업을 수행하고 있다.

1982년 9월, ISO에서는 날로 고도화하고 복합화하는 산업 자동화 시스템의 효과적인 표준화를 지원하기 위하여 새로운 기술위원회의 설립을 결정하였고, 이에따라 1983년에 ISO TC 184의 제1차 총회가 개최되었다. 또한 ISO와 IEC의 작업영역의 조정을 위한 ISO/IEC Joint Technical Programming Committee(JTPC)는 수차례의 토의를 거쳐, TC 184의 명칭과 범위를 다음과 같이 정하였다.

◆ 명칭 : 산업자동화 시스템 (Industrial Automation System)

◆ 범위 : 정보시스템, 기계장비 및 통신들과 같은 복합기술의 응용을 포괄하는 개별 소자 생산(discrete part manufacturing)에 관한 산업자동화 시스템의 표준화

그후 산업자동화에 있어 통신의 중요성이 크게 부각됨에 따라, 1989년에는 명칭을 Industrial Automation and Communications System으로 改稱하였고 취급범위도 "discrete part manufacturing에 관한 산업자동화와 그와 관련된 통합분야에 있어서의 표준화"로 변경되었다.

ISO/TC 184에는 IEC/TC 44에서 다루는 전기, 전자장치, IEC/TC 65에서 다루는 일반 응용을 위한 프로그래머블 로직 콘트롤러는 제외된다.

◆ IEC/TC 65 : 공업-프로세스 계측과 제어(Industrial-process measurement and control) 연속 공정 및 프로세스(continuous and batch process)에 관한 산업공정의 계측과 제어에 사용되는 시스템과 부품에 관한 국제표준을 정함.

- ◆ IEC/TC 44 : 산업기계의 전기장치(Electrical equipment of industrial machines)전기, 전자 장치와 시스템의, 작업중에 손으로 운반할 수 없는 공업기계(고위 계층 시스템 관점이 제외된, 연관 동작을 하는 기계들의 그룹을 포함하는)의 응용에 주로 관련된 국제 표준의 제정.

TC 184는 표결 사항에 대해서 투표권을 갖는 17개의 P-멤버와, 투표권이 없는 19개의 O-멤버 국가로 구성되어 있으며, 간사 기관은 프랑스의 AFNOR(Association Francaise de Normalisation)이다.

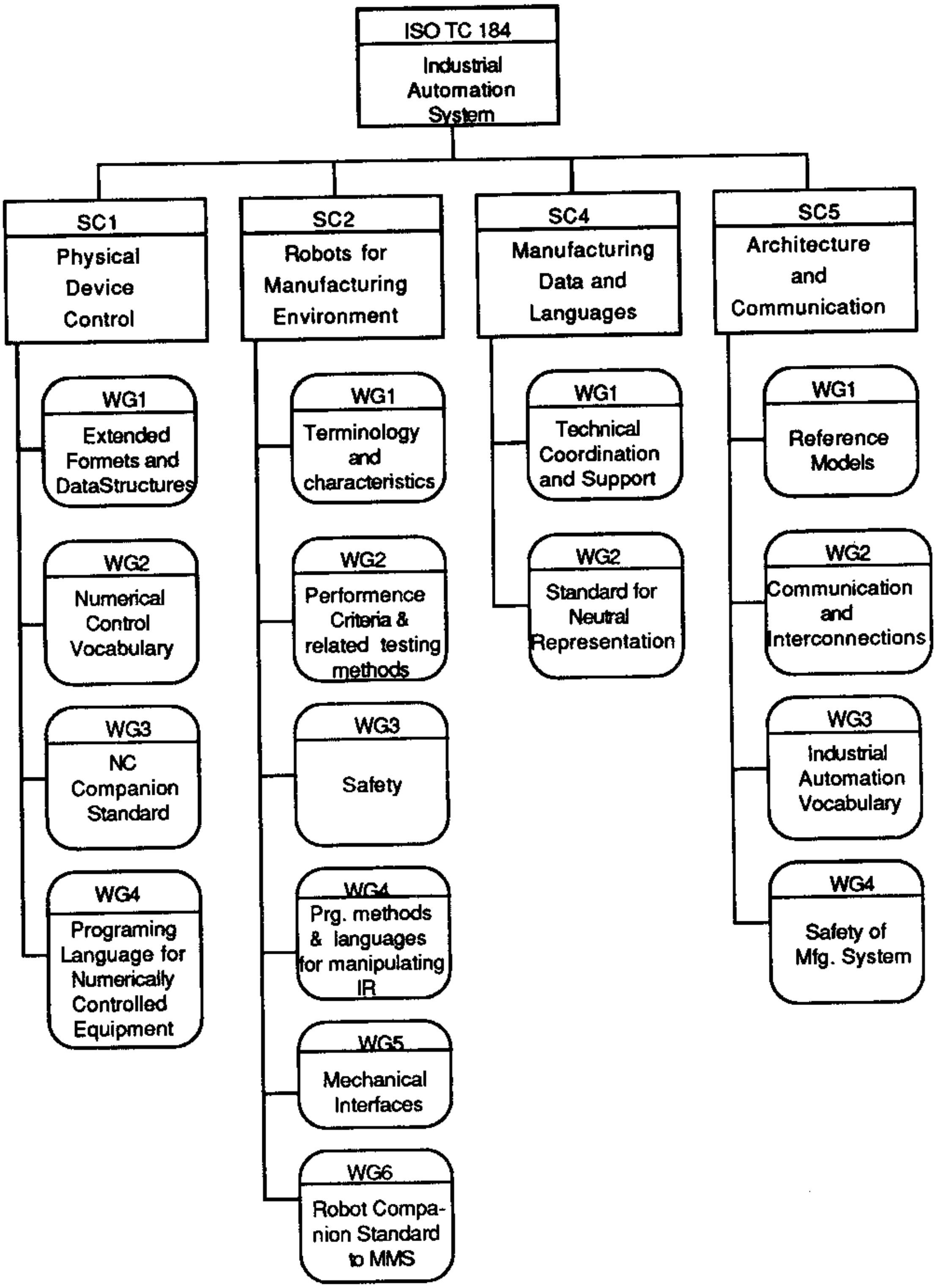
TC 184는 산업자동화 시스템의 궁극적 목표를 CIM(Computer Integrated Manufacturing)으로 보고 있어, CIM을 실현하는데 필요한 여러 기술 요소들을 개념적으로 분류하여 그에 해당하는 부위원회(SC)와 워킹 그룹(WG)을 두고 있다. 1991년 현재 TC 184는 4개의 SC와 16개의 WG으로 구성되어 있으며, 그 구조는 <도16>과 같다.

1. SC1-Physical Device Control

NC머신 등 장비의 코드, 포맷, 축, 동작 용어, 데이터 구조, 명령 언어 그리고 이에 관련된 시스템 측면을 표준화한다.

- ◆ WG1 - Extended Format and Data Structure

  데이터와 프로세스 파라미터의 변경을 가능케하는 실시간 응용의 요구 사항들을 커버한다. 이것은 확장된 포맷과 데이터 구조를 통하여 구현되어야 한다. 정보교환을 위한 응용 계측도 커버한다.


ISO TC 184
Industrial Automation System
SC1
Physical Device Control
WG1
Extended Formets and DataStructures
WG2
Numerical Control Vocabulary
WG3
NC Companion Standard
WG4
Programing Language for Numerically Controlled Equipment
SC2
Robots for Manufacturing Environment
WG1
Terminology and characteristics
WG2
Performence Criteria & related testing methods
WG3
Safety
WG4
Prg. methods & languages for manipulating IR
WG5
Mechanical Interfaces
WG6
Robot Compa-nion Standard to MMS
SC4
Manufacturing Data and Languages
WG1
Technical Coordination and Support
WG2
Standard for Neutral Representation
SC5
Architecture and Communication
WG1
Reference Models
WG2
Communication and Interconnections
WG3
Industrial Automation Vocabulary
WG4
Safety of Mfg. System

<도16> ISO/TC 184의 구성

◆ WG2 - Numerical Control Vocabulary

TC 184에서 커버되는 NC머신과 장치에 관한 일반 어휘와 특수 어휘를 확립한다.

◆ WG3 - NC Companion Standard

MMS(Manufacturing Message Specification)에 부수되는, NC 머신용의 표준을 개발한다.

◆ WG4 Programming Language for Numerically Controlled Equipment

NC 프로세스의 입출력을 취급하는 국제표준을 제시하고 개정한다.

2. SC 2 - Robots for Manufacturing Environment

다음과 같은 항목을 표준화한다.

- 정의, 특성
- 용어
- 그래픽 표현(Graphic Representation)
- 성능 및 성능 테스트 방법
- 안전성
- 기계적 인터페이스
- 프로그래밍 방법
- 정보 교환을 위한 요구 사항

◆ WG1 - Terminology and Characteristics

SC 2 범위내에서 로보트의 정의, 특성화, 용어, 그래픽 표현을 표준화한다.

◆ WG2 - Performance Criteria and related testing methods

사용자에게 로보트에 관한 충분한 스펙을 제공하고, 표준화된 방법으로 그것을 확인할 수 있도록 하는 성능기준과 그에 관계된 테스트 방법을 표준화한다.

◆ WG3 - Safety

SC2 범위내에서 로보트의 설계, 제작, 설치, 사용 및 유지보수에 있어서의 안전성에 관한 표준을 제공한다.

◆ WG4 - Programming methods and languages for manipulating industrial robot

여러 레벨의 프로그래밍 방법과, 산업용 로보트를 조작하기 위한 언어의 표준을 개발한다.

◆ WG5 - Mechanical Interface

기계적 부착, munipulating 산업 로보트와 그 엔드 이펙터의 수동 또는 자동의 교환성을 보장하기 위한, 표면 장착 또는 잠금(locking) 장치의 형태와 기본적인 디멘젼을 표준화한다.

◆ WG6 - Robot Companion Standard to MMS

MMS에 수반되는, 로보트에 있어서의 표준을 개발한다.

3. SC 4 - Manufacturing Data and Languages

생산품의 Life Cycle 전체를 통하여 완전성과 Integrity의 손실이 없이 전산화된 Product Model 정보의 획득을 가능케 하는 표준과, 자동화된 생산기계 및 시스템의 프로그래밍 언어표준을 제정한다.

◆ WG1 - Technical Coordination and Support

- SC4에 기술적인 지원과 권고안을 제공한다.

- 국가간의 기술개발을 조정한다.

- 기술적인 차이점을 해결한다.

- ISO 문서의 초안에 포함될 국가들의 기고서를 종합한다.

◆ WG2 - Standard for Neutral Representation

제품정보(Product Model)의 컴퓨터 표현을 neutral하게 할 수 있도록 표준을 정한다.

4. SC5 - Architecture and Communication

시스템통합과 생산환경의 통신 및 상호접속을 가능케하는 표준에 대한 요구사항을 찾아낸다. 이를 위하여

1) 생산품의 고안에서부터 분배에 이르기까지, 생산환경의 자동화 영역에 있어서 시스템 통합을 위한 Reference Model 및 Architecture를 정의한다.

2) 생산 자동화용 통신과 기기 상호접속을 위한 표준의 요구사항을 고려하며 현존하는, 생산 환경에 적합한 통신 규격을 선정하거나 개발한다.

3) 여러 TC 184 단체들의 어휘 활동을 조정한다. 확정된 표준 어휘를 배출하고 그것을 영속적으로 유지한다.

◆ WG1 - Reference models

시스템 자동화 요소간의 각종 인터페이스와 특성(예를 들어, 전기, 기계, 맨-머신, 정보, 순서, 언어 등)의 정의를 통하여, 표준화를 위한 장기 계획을 돕는 밑받침이 될 수 있는 다차원의 공개된 참조 모델을 만들거나, 기존의 작업 결과를 조사한다. 최소한 다음과 같은 타입의 모델을 고려한다.

i) 테스크 그룹과 그들간의 관계로 정의되며, data-orient될 수도

있는, 공장(또는 회사)을 나타내는 기능 모델 (functional model).

ii) 다음과 같은 측면을 나타내는 다른 모델

- 데이터 구조
- 데이터 베이스 구성
- 통신
- 전기적/물리적 인터페이스

◆ WG2 - Communication and Interconnections

자동화된 공장에 있는 컴퓨터, 장비, 시스템을 상호접속하기 위한 인터페이스, 프로토콜, 데이커 포맷, 멧시지 구조를 표준화한다.

이 장비들에는 NC머신, 산업용 로보트, 프로그래머블 콘트롤러, 계측 장비 그리고 자동화를 위한 다음 장치들이 포함된다.

◆ WG3 - Industrial Automation Vocabulary

산업자동화용 어휘를 개발하고 유지시킨다.

◆ WG4 - Safety of Manufacturing System

산업자동화에 있어서 생산시스템의 안전성을 위한 표준을 정한다.

제3절 국내 동향

CIM기술은 3차산업혁명(정보혁명)이라 불리울 만큼 세계경제체제까지도 크게 변모시킬 수 있는 잠재력을 갖고 있어, 일본의 경우 ISO/TC 184의 모든 SC에 주 member로 가입하고 각 SC에 수백개의 회사가 참여하여 활동하고 있다. 우리나라는 TC184에만 P-Membership을 갖고 있을

뿐, SC 가입은 全無하여 활동이 매우 저조한 실정이다. 산업자동화 및 통신 분야에 대한 연구개발 및 표준화 활동은 선진국에서도 비교적 초기단계에 있는 편이므로, 우리나라로서도 이젠 체계적이고 효율적인 국내조직을 갖추어 적극적인 대응 전략을 세우는 것이 필요한 시점이다.

이를 위하여 1991년 2월에 공업진흥청에서 담당자와 KIST, ETRI가 함께 모여 국내전문위원회 구성을 위한 예비 모임을 가진 후, 전문위원 선정작업을 거처, 1991년 11월에 전문위원회가 정식으로 발족되었다. ISO/TC 184 국내전문위원회의 위원구성은 <표8>과 같으며, 향후 SC부문별로 하부조직을 구성하고 SC member로서 표준화 활동에 참가하는 등 국내의 CIM표준화를 선도하는 많은 활동이 기대된다.

성명	소속	비고
권욱현	서울대학교	의장
한장섭	공업진흥청	간사
황승구	전자통신연구소	운영간사
강무진	한국과학기술연구원	
권문식	현대정공	
국금환	한국기계연구소	
노형민	한국과학기술연구원	
박삼진	한국기계연구소	
서승완	한국해사기술연구소	
임계영	금성산전	
정명진	한국과학기술원	
조증성	한국표준과학연구원	

<표8> ISO/TC 184 국내전문기술위원회 위원

(참고 문헌)

1. CIM Architecture, CAM-I report R-88-ATPC-01, 1988.
2. A Reference Model for Computer Integrated Manufacturing (CIM) - A Description from the viewpoint of Industrial Automation, CIM Reference Model Committee, 1983.
3. Design Rules for a CIM System, ESPRIT-CIM 5, 1/34, Elsevier Science Publishers, 1986.
4. Overview of CIM activities in the Europen Community, P. Mc. Connail, Proc. MSTF'91, Twente, 1991.
5. Results of the Eureka-FAMOS research programme. A Europen contribution in manufacturing technology, F. Jovane, Proc. MSTF '91, Twente, 1991.
6. 실천 CIM구축법, 東正則, 공업조사회 (日), 1991.
7. 인간 Support CIM 구축법, 根津和雄, 공업조사회 (日), 1991.
8. MS 국제공동프로그램에 대하여, FURUKAWA, 일본기계학회지, 1991. 3.
9. CIM구축과 국제화, UWATOKO, 일본기계학회지, 1991. 3.
10. CIM Management, VDI-GACIM, 1990.
11. CIRP Annals Manufacturing Technology, 1990.
12. Japanese Proposal and Progress Report on IMS (Draft), IMS Promotion Center, 1991.
13. The Historical Evolution and Future Perspectives of CIM - A New Concept of I-CIM, I. Ham, 1991.

14. IMS-Joint International Research Programs into an Intelligent Manufacturing System, IMS Promotion Center, 1990.

15. IMS 국제공동연구프로그램의 개요, 통상산업성(日), 기계진흥, 1990.

16. IMS센타의 설립과 금후의 전망, 국제로보트.FA기술센타(日), 기계진흥, 1990

17. EC의 FA연구개발계획, 大見 孝吉, 기계진흥, 1990.

18. BRITE/EURAM Programme, Commission of the European Communities, 1990.

19. Integrating the Automated Factory, SME, 1988.

20. PDES : The Enterprise Data Standard, SME, 1989.

21. Enterprise Information Exchange (EIX) Issues in the CIM Environment, SME, 1991.

22. CIM Integration Tools, SME, 1988.

23. CIM-A Working Definition, SME, 1990.

24. Integrations management, IAO-Büroforum, 1989.

25. CIM-生販統合의 실현, 油井兄朝, 일본경제신문사, 1990.

26. CIM최신공장사례, 中村康, Proc. 2nd CIM Japan Seminar, (R-33) 1991.

27. 2000년을 향한 Global기업의 엽무혁신, 油井直次, Proc. 2nd CIM Japan Seminar (S-4-A), 1991.

28. OSI와 FAIS의 개발, Proc. 2nd CIM Japan Seminar, (R-14), 1991.

29. Open System Architecture for Computer Integrated Manufacture, ESPRIT nr 688, 1987.

30. Status of PDES-Related Activities - Standards & Testing, NIST

(NISTIR 4432), 1990.

31. Framework for CIM System Integration, DIN-NAM 96.5 Proposal for TC184/ SC5, 1988.

32. Reference Model for Shop Floor Production Standards, ISO/TC184 N147 (Technical report 10314), 1989.

33. The Joint International Research Program into Intelligent Manufacturing Systems - Part I : Overview of the Initial Program, Part II : The Reaction from the Europe and North America and How it changed the Program's Framework, FURUKAWA, 1991.

(부록1)

Framework for CIM system integration

DIN-NAM 96.5 Proposal

for New Work Item
in TC 184 SC5
on a

Framework for CIM System Integration.

TABLE OF CONTENT.

1.0 Introduction.
- 1.1 About this Framework.
- 1.2 About the Need for this Framework.

2.0 Scope and Field of Application.

3.0 References.

4.0 Abbreviations.

5.0 Definitions.

6.0 Architectural Framework.
- 6.1 Definitions.
- 6.2 Levels of Genericity.
- 6.3 Levels of Modelling.
- 6.4 Levels of Views.
- 6.5 Summary of Framework.

7.0 Particular Enterprise Modelling
- 7.1 Definitions.
- 7.2 Enterprise Modelling Level.
  - 7.2.1 Enterprise Function View.
  - 7.2.2 Enterprise Information View.
  - 7.2.3 Enterprise Resource View.
  - 7.2.4 Enterprise Organisation View.
- 7.3 Intermediate Modelling Level.
  - 7.3.1 Intermediate Function View.
  - 7.3.2 Intermediate Information View.
  - 7.3.3 Intermediate Resource View.
  - 7.3.4 Intermediate Organisation View.
- 7.4 Implementation Modelling Level
  - 7.4.1 Implementation Function View.
  - 7.4.2 Implementation Information View.
  - 7.4.3 Implementation Resource View.
  - 7.4.4 Implementation Organisation View.
- 7.5 Summary.

8.0 Impact on Standardisation.

Appendix A: CIM Support Systems.
- A.1 Definitions.
- A.2 The Integrated Processing Environment.
- A.3 The Integrating Infrastructure
  - A.3.01 The Human Front End Service.
  - A.3.02 The Machine Front End Service.
  - A.3.03 The Application Front End.
  - A.3.04 The Data Management Front End
  - A.3.05 The Communication Front End.
  - A.3.06 The Activity Control Service.
  - A.3.07 The Business Control Service
  - A.3.08 The Resource Management Service.
  - A.3.09 The System Wide Data Service.
  - A.3.10 The System Wide Exchange Service.

1.0 INTRODUCTION.

1.1 About this Framework.

The purpose of this framework is to provide a commom basis for identifying and coordinating standards development for the purpose of CIM Enterprise Modelling and CIM Systems Integration, while allowing existing standards to be placed into perspective within this framework.

It is not the intent of this framework either to serve as an implementation specification, or to be a basis for appraising the conformance of actual implementations. Rather this framework provides concepts which allow international teams of experts to work productively and independently on the development of unique standards of parts of the total framework.

The justification for development of standards shall follow normal administrative procedures even though the need for such standards can be assumed from this framework.

The description of the framework is developed in stages:

Clause 6 describes three different framework levels:
levels of genericity,
levels of modelling,
levels of views.

Clause 7 describes the enterprise modelling process, i.e. the sequential steps to be performed to achieve a computer processable description (the model) of an enterprise. The last step of this modelling process is the transformation of the model into the real world.

Clause 8 makes suggestions in which field standards may be developed to support the framework.

Appendix A discusses briefly a Information Technology environment suggested, to support the generation of a computer processable enterprise model (the build-time) and to support the Information Technology environment for implementations built according to this framework (the run-time). This appendix is for information only and will be transferred into a separate standard after specifications of this environment are more precise.

1.2 About the Need for this Framework.

The manufacturing process, i.e.the transformation of raw material into marketable products is changing from a semistable process to a highly dynamic one. The reasons for this change are manifold and only some of the major reasons are listed in the following:

Fast changes in market demands leads to fast obsolescence of established products.

World-wide availability of technology, capital and information (know-how) leads to shorter development cycles for new products.

World-wide marketing of products leads to strong cost competition in established markets.

The result for the manufacturing industry is an upcoming era of permanent change in its economic and technology environment. To cope with this permanent change is the major future challenge for the manufacturing industry. For each company the challenge is to stay synchronized with the external changes; i.e. to have an adaptation cycle shorter than the external one.

Today the manufacturing industry still strives for stability of its production as a major enterprise goal. Therefore, management of change is not yet considered a permanent objective in the manufacturing industry.

Information processing is still very much fragmented even in computerized applications. This is due to past bottom up generation of computer applications without the use of a general framework, to the use of multi-vendor hardware and software and to the organizational boundaries in the companies. Therefore, the decision making process in the companies is still based on traditional information processing - information gathering with 'paper and pencil', on request and from inconsistent sources. This process is at the least very time consuming. In many cases it yields only insufficient or even outright wrong information.

In addition, the companies are not organized for fast decision making processes. Departments are still managed according to their own sub-goals rather than to real enterprise goals. The responsibilities are still structured in one-dimensional hierarchies which mix responsibilities for enterprise assets with those for enterprise operations. Matrix organization is still a theoretical concept.

It is expected that CIM will improve enterprise competitiveness through adaptability and flexibility of enterprise operation and organisation. CIM shall ensure efficient use of enterprise assets and resources like people, capital investments and information.

To reach this goal information technology provides a very important and powerful tool for the enterprise. However, a tool which still needs further improvements before it will fulfil the high expectations. It is the intend of this framework to define many of the improvements required for reaching the high goals of CIM.

Information Technology hardware makes it possible even today to install very large networks of computer systems with almost unrestricted performance and processing capabilities. What is missing is the necessary software which allows a meaningful processing of the vast amount of information in the manufacturing enterprise. This meaningful processing, i.e. the processing of the right information, for the right purpose, at the right time, in the right place is still the major problem in Information Technology application.

Todays computer application programs have been developed as standalone units. Functional aspects, information and data aspects and even organisational aspects are part of the application thus portability to other enterprises is limited even if heterogeneous Information Technology Systems would allow.

Application software and its development are not the only weak points in the application of information technology. Another one is the problem of information integration, i.e. the ability to access information generated and used by different applications, running on different computer systems.

To cope with the permanent change of the environment the enterprise has to be able to manage in 'real time' the required internal changes. That means to keep the internal adaptation shorter than the external change cycle. This requires the 'real time' control of the whole manufacturing process from material source to product service in the market. It requires the ability to make decisions in 'real time' as well.

To establish the required fast decision making process all responsibilities in a company must be visible. This will not be the equivalent of todays organization chart but will be the explicit knowledge about all decision making individuals in the company and about their explicit responsibilities for processes and resources.

To support the decision making process the right information has to be accessible in time and the results of the decision making process have to be available in the right place. Decision making also has to take into account alternative solutions and their possible impacts on the total enterprise operation. Therefore, processable description and simulation of alternative solutions is a requirement on future decision support systems.

Integration is expected to solve many of the problems in the manufacturing industry today. However, integration is understood in different ways by the people in the industry. The users expect CIM to provide integration of information in terms of availability, accessibility and consistency. The vendors see system integration in terms of hardware system connectivity and portability of computer application as the main goal of CIM. Therefore integration has to be done in more than one aspect. There are four levels of integration which have been identified. The Physical System Integration is mainly concerned with intersystem communication. It is expected that this level of integration will be provided by current Information Technology concepts and standards (e.g. OSI). The other three levels of integration are Information Integration, Application Integration and Business Integration. These aspects will specifically be addressed in this framework.

2.0 SCOPE AND FIELD OF APPLICATION.

The goal of this framework is to provide a means of describing enterprises in a processable description which will make the forthcoming permanent changes in the business environment manageable. Starting from the business needs, all aspects of the information required by all functions in a manufacturing enterprise will be covered.

A framework for modelling an enterprise and developing a computer supported implementation of the model on homogeneous as well as heterogeneous Information Technology Systems will be defined. This means CIM systems developed and constructed according to the concepts of this framework will support all levels of management in their strategic, tactical and operational planning as well as the direct operation at the shop floor. These CIM systems will allow to control and to monitor all actions carried out in the enterprise. They will support all decision making processes by providing the required information and by allowing simulation of alternatives and optimisation of solutions before implementation.

The framework will be applicable to new CIM system designs as well as to the integration of existing CIM systems, graceful migration is an important part of the goals.

3.0 REFERENCES.

All the information contained in this document has been obtained from the CIM-OSA Reference Architecture Specifications developed in the ESPRIT AMICE project.

4.0 ABBREVIATIONS.

AMICE	European CIM Architecture
ESPRIT	European Strategic Program for Research and Development in Information Technology
IIS	Integrating Infrastructure

5.0 DEFINITIONS.

Definition of terms are included at the beginning of individual clauses and subclauses.

6.0 ARCHITECTURAL FRAMEWORK.

6.1 Definitions.

Architectural Levels: A defined part of the total Enterprise Modelling Process. Three architectural levels are defined Generic Level, Partial Level and Particular Level. Each architectural level covers the three Modelling Levels and four different views.

CIM System: A CIM System is the means by which enterprise tasks are performed in the most effective and economic way possible, using Information and Manufacturing Technology where ever useful and possible.

Enterprise: An entire corporation, consisting of one or more organisational components, with the prime objective of producing products or offering service.

Enterprise Modelling Level: The Enterprise Modelling Level describes in a business sense and terminology "what" has to be done. This is the field of activity of the user and his business requirements. This still leaves options for the implementation to be chosen at both the Intermediate Modelling Level and the Implementation Modelling Level.

Enterprise Views. Within the Enterprise Modelling Levels the enterprise is described in terms of views. Four views are identified :

Function View - the functional structure
Information View - the information structure
Resource View - the resource organization
Organization View - the organizational structure

Function Model. All the aspects of the Function View generated during the enterprise modelling process result in the Function Model.

Function View. The Enterprise Function View provides a description of the functional structure of the CIM System requirements, in terms of the enterprise objectives. The view has a relationship with the outside world since it also reflects the constraints and recognises the relevant inputs and outputs.

Generic Architecture Level. The Generic Architecture Level is a reference catalogue of basic architectural constructs (building blocks) for components, constraints, rules, terms, services, functions and protocols. Constructs described at this level have the widest application in CIM.

Information Model. All the aspects of the Information View generated during the enterprise modelling process result in the Information Model.

Information View. The Information View provides a description of the information structure of the CIM System requirements. The view has a relationship with the outside world since it also reflects the constraints and recognises the relevant inputs and outputs.

Implementation Modelling Level. The Implementation Modelling Level describes the "means" for executing the processes and activities. The Implementation Modelling Level specifies an integrated set of components necessary for effective realization of the enterprise operations.

Intermediate Modelling Level. The Intermediate Modelling Level describes "how" the processes and activities are performed. This still leaves options for the means of the execution to be defined at the Implementation Modelling Level. The Intermediate Modelling Level also structures and optimises the business requirements defined in the Enterprise Modelling Level by the multiplicity of users according to overall business and CIM System constraints.

Modelling Level. A Modelling Level is a defined part of the total Enterprise Modelling Process. Three Modelling Levels are defined, the Enterprise Modelling Level, the Intermediate Modelling Level and the Implementation Modelling Level. Each Modelling Level covers the three Architectural Levels and consists of the four defined Enterprise Views.

Organisation Model. All the aspects of the Organisation View generated during the enterprise modelling process result in the Organisation Model.

Organisation View. The Organisation View provides a description of the organisational structure of the CIM System requirements.

Partial Architecture Level. The Partial Architecture Level is concerned with sets of partial instantiated models applicable to a defined category of manufacturing enterprises or processes.

Particular Architecture Level. The Particular Architecture Level is entirely concerned with one particular enterprise. For such an enterprise the CIM System implementation has to be described by a Particular Architecture. This architecture embodies all necessary knowledge of the enterprise in a form which can be used directly for the specification of an integrated set of Manufacturing and Information Technology components. These components comprise the implemented CIM System which satisfies the business requirements of that particular enterprise.

Resource Model. All the aspects of the Resource View generated during the enterprise modelling process result in the Resource Model.

Resource View. The Enterprise Resource View provides a description of the resource organisation of the CIM System requirements. The view has a relationship with the outside world since it also reflects the constraints and recognises the relevant inputs and outputs.

Stepwise derivation. The process of deriving from Enterprise Modelling level via Intermediate Modelling level to Implementation Modelling level.

Stepwise generation. The process of generating Enterprise Views including requirements gathering and its content in terms of function requirements and generating all views required in an iterative optimisation manner.

Stepwise instantiation. The process of orderly movement from a Generic Architectural Level via a Partial Architectural Level to the Particular Architectural Level.

.2 Levels of Genericity.

The CIM Reference Model has three levels of architectural genericity (Figure 6-1), the Generic, the Partial and the Particular Levels. The Generic and the Partial levels are two parts of a Reference Architecture. These two levels describe the constructs required to gather the user requirements for a CIM System operation, to build a model of an enterprise and to derive from this model a consistent CIM System description and implementation. These two architectural levels are independent from a specific enterprise, that means the constructs and descriptions that will be provided over time are abstract.

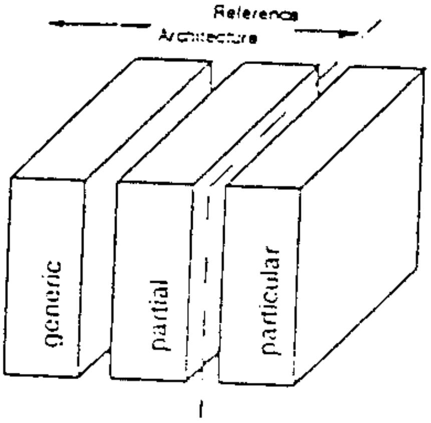


Figure 6-1 Levels of Genericity.

The Generic Level is a reference catalogue of basic architectural constructs (building blocks) for components, constraints, rules, terms, service functions and protocols. Constructs described at this level have the widest application in CIM.

The Partial Level is concerned with sets of partial instantiated models applicable to a particular category of manufacturing enterprises, processes or domains. Such a partial instantiation will consist of typical structures for a variety of categories like industry-sector types (such as aerospace, automotive, electronics, etc.,) company size, national variations, or typical structures for typical processes like Enterprise Procurement or Computer Assisted Product Design. Partial models may also be defined in a hierarchy (Figure 6-2). A set of partial models 'Automotive Suppliers' may be partially instantiated in a further subset according to enterprise size and even in a further subset according to type of business. Requirements for these structures will grow with time. Therefore, the Partial Level is considered to be an open set. This set will be populated according to the needs of standardization bodies, industry offerings and even internal work in particular enterprises. The partial models are the prime means by which the Reference Model encapsulates industry requirements, and provides a more realistic and usable tool for a particular enterprise.

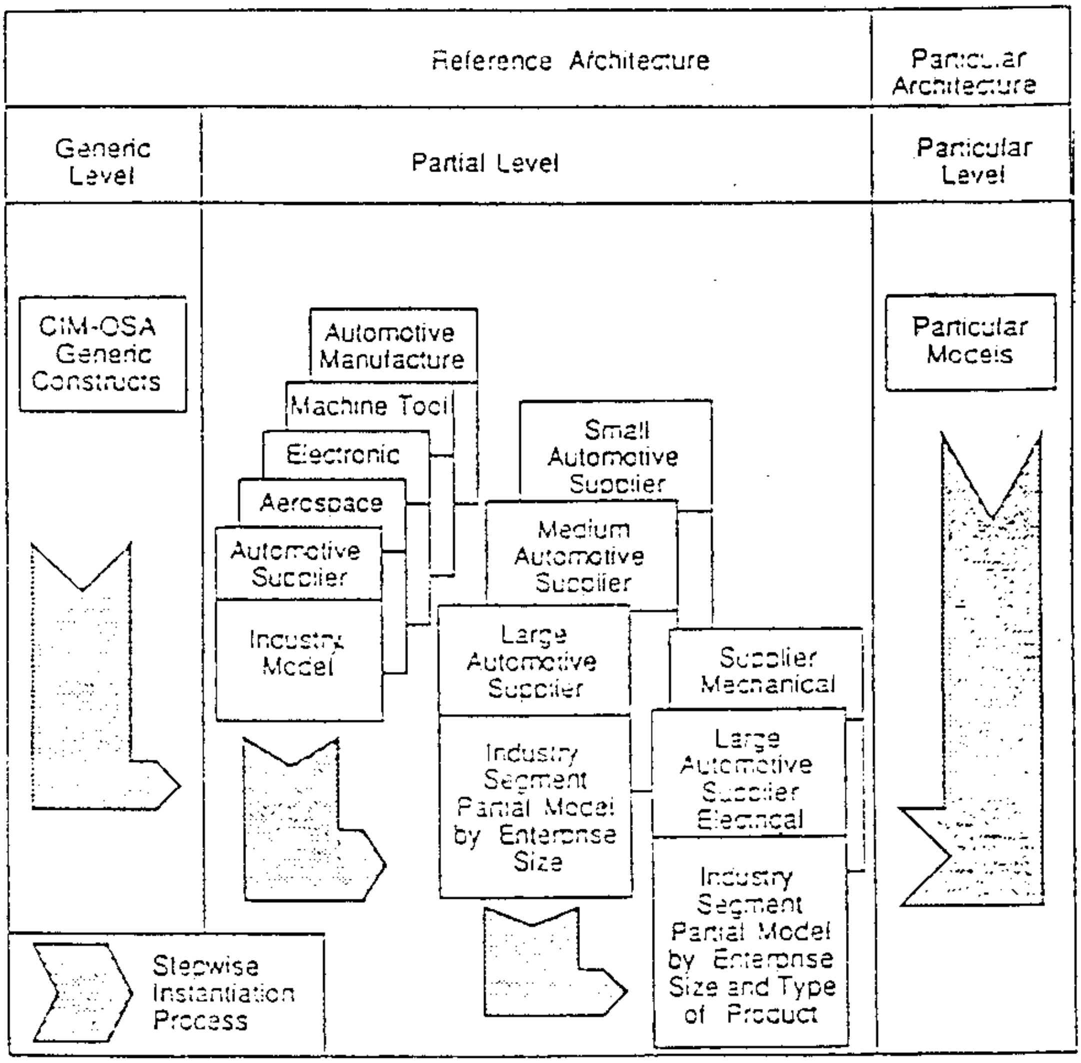

Reference Architecture
Particular Architecture
Generic Level
Partial Level
Particular Level
CIM-OSA Generic Constructs
Automotive Manufacture
Machine Tool
Electronic
Aerospace
Automotive Supplier
Industry Model
Small Automotive Supplier
Medium Automotive Supplier
Large Automotive Supplier
Supplier Mechanical
Large Automotive Supplier Electrical
Industry Segment Partial Model by Enterprise Size
Industry Segment Partial Model by Enterprise Size and Type of Product
Particular Models
Stepwise Instantiation Process

Figure 6-2 Partial Models.

> Note: A typical Partial Model is the MAP/TOP Specification, which is instantiated from the Generic "ISO Basic Reference Model for Open Systems Interconnection (ISO IS 7498)".

The Particular Level is entirely concerned with one particular enterprise. For such an enterprise the CIM system implementation has to be described by a Particular Architecture. This architecture embodies all necessary knowledge of the enterprise in respect to requirements and behaviour, in a form which can be used directly for the specification of an integrated set of Manufacturing and Information Technology components. These components comprise the implemented CIM System which satisfies the business requirements of that particular enterprise. The Particular Level can be instantiated either directly from the Generic Level and/or from the Partial Level.

This stepwise instantiation process permits the orderly movement from a more Generic Level to a more Particular level. A number of steps performed in the correct order permit movement from generic, abstract constructs to completely determined particular CIM Systems. The process of stepwise instantiation will be supported, over time, by knowledge based systems, which assist the enterprise designer in defining the processes and activities as well as the data and resources required for them..

6.3 Levels of Modelling.

The Reference Model provides three Modelling Levels, the Enterprise Modelling Level, the Intermediate Modelling Level and the Implementation Modelling Level (figure 6-3).

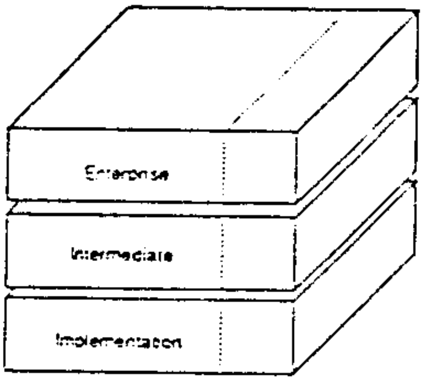


Figure 6-3. Levels of Modelling.

At the Enterprise Modelling Level the business requirements of the enterprise are identified. The identified requirements lead to the definition of enterprise processes, their inputs and results, and the required activities within these processes.

Since these processes will be defined independently from each other by different people being knowledgeable on only some of the processes it is unavoidable that redundancies and suboptima are defined. These short-comings will be removed at the Intermediate Modelling Level to ensure the most economic and flexible operation possible.

The Intermediate Modelling Level will be used. to further detail the business processes and activities by describing how they are performed. Suboptima and redundancies will be removed, volumes will be added to the models, executing entities will be specified and constraints provided by the overall enterprise and/or the external world will be considered. The Intermediate Modelling Level also serves as an isolation between the Enterprise Modelling and the Implementation Modelling Level. This will enable to re-investigate trade-offs and or decisions made due to internal or external constraints.

The Implementation Modelling Level defines the means of execution by selecting the real entities (humans, machines and programs) being used to execute the processes and activities defined in the Enterprise Modelling Level. The entities will be selected according to the specifications defined at Intermediate Modelling Level from a catalogue of components offered by various suppliers. Components required, but not available on the market must be developed by the particular enterprise.

The process of stepwise derivation from the requirements defined at the Enterprise Modelling Level through the Intermediate Modelling Level towards the real component selection at the Intermediate Modelling Level will be assisted over time by a set of computer programs, which offer alternative solutions, identify missing or inconsistently defined objects, processes or activities. Although the stepwise derivation process is described in a one-way direction it may be necessary for individual processes or activities to reconsider the previous Modelling Level.

6.4 Levels of Views.

Four different views (Figure 6-4) have been identified which allow to model the major aspects of the enterprise independent of each other.

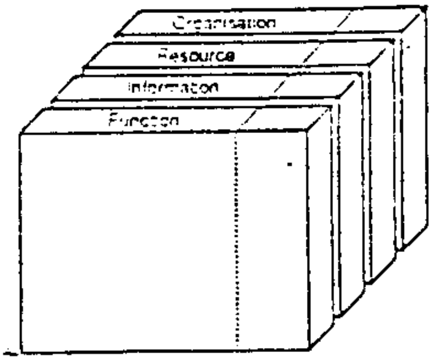


Figure 6-4. Levels of Views.

The Function View is the representation of the enterprise operation in terms of a set of hierarchically structured business processes. Each business process is defined by its triggering events, the results it produces and by its explicit control flow description (its procedural rule set). The internal structure of a business process controlled by the procedural rule set may be either lower level business processes or executable Enterprise Activities. The Enterprise Activities may control by procedural rules the Functional Operations, which are executed by Functional Entities. All the aspects of the Function View generated during the enterprise modelling process result in the Function Model.

The Information View gathers all information defined and contained in the enterprise. The information is structured through a hierarchically defined set of information classes and by objects, object views and object view editions within domains to be defined by the enterprise designer. Object, views and editions can be broken down into information entities, which are the smallest addressable information items. All the aspects of the Information View generated during the enterprise modelling process result in the Information Model.

The Resource View contains all relevant information on enterprise resources. The view is structured using the hierarchical concept of cells for grouping resources according to enterprise requirements. There is a strong relationship to the Functional Entities defined in the Implementation Function View. All the aspects of the Resource View generated during the enterprise modelling process result in the Resource Model.

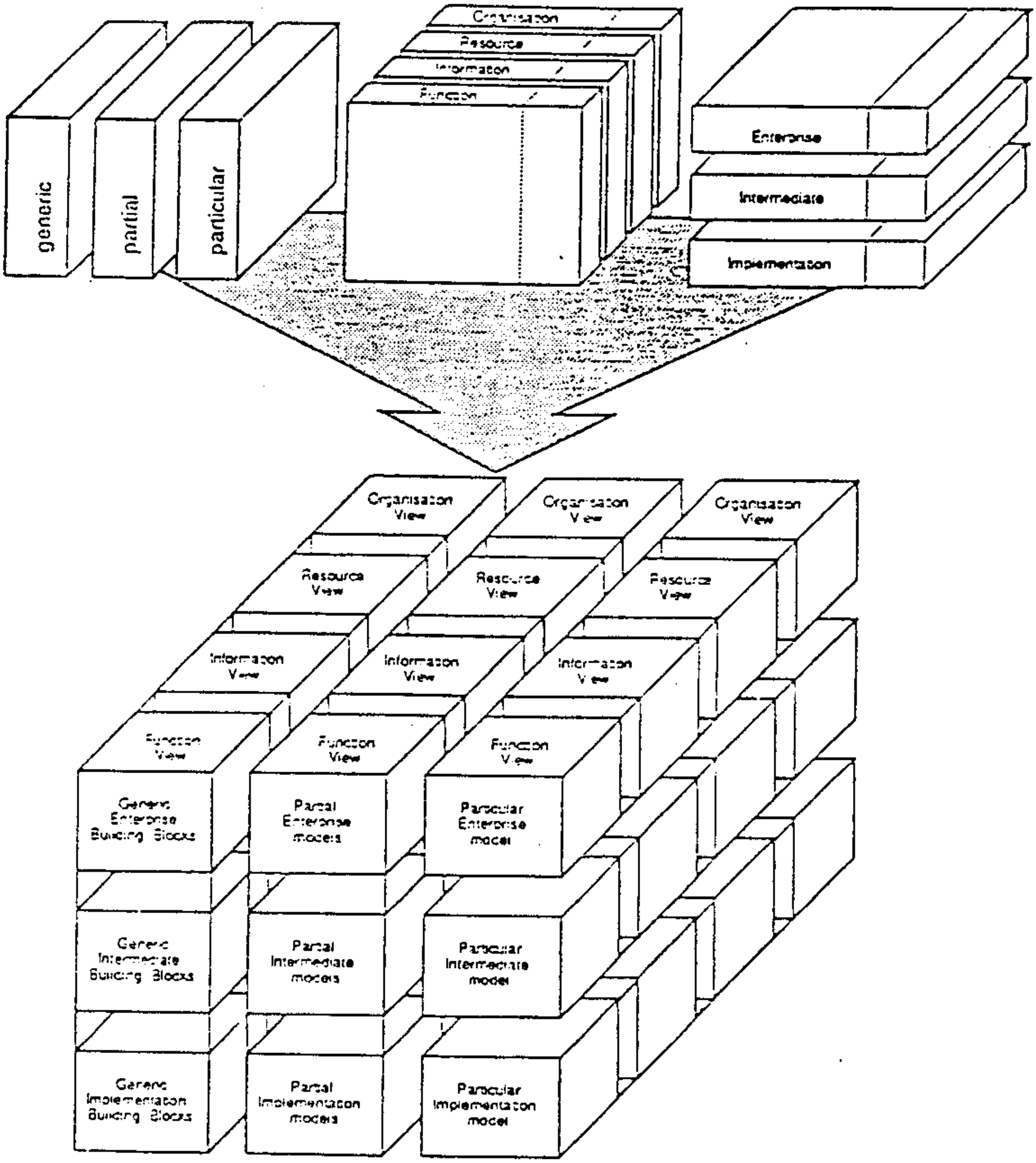
The Organisation View contains all the relevant information on the responsibilities within the enterprise and allows to gather and structure the different responsibilities in the enterprise (for functions, information and resources). The view is structured using the hierarchical concept of cells for grouping the organisational responsibilities according to the enterprise requirements. All the aspects of the Organisation View generated during the enterprise modelling process result in the Organisation Model.

Each of the four views is comprised of the related contents of all the architectural and modelling levels. The views will be generated one after each other and the process of this stepwise generation will be supported over time by a set of application programs, which support the enterprise designer with procedures how to move forward and partial models and identifies inconsistencies and missing items during the modelling process. Although the stepwise generation process is described in a one-way direction it may be necessary for individual processes or activities to reconsider the previous views.

6.5 Summary of Framework.

Figure 6-5 shows the total framework resulting from the overlay of all architectural and modelling levels with all views.

Only the Particular Architecture Level is to be provided by the designer of a particular enterprise, as he has to generate the views for his particular enterprise and to derivate from the requirements (Enterprise Level) to the real implementation. The Generic Architecture Level will ensure compatible and portable business descriptions, while the Partial Architecture level will provide over time the designer the necessary partial solutions to ease the complex work of enterprise modelling.



generic
partial
particular
Organisation
Resource
Information
Function
Enterprise
Intermediate
Implementation
Organisation View
Organisation View
Organisation View
Resource View
Resource View
Resource View
Information View
Information View
Information View
Function View
Function View
Function View
Generic Enterprise Building Blocks
Partial Enterprise models
Particular Enterprise model
Generic Intermediate Building Blocks
Partial Intermediate models
Particular Intermediate model
Generic Implementation Building Blocks
Partial Implementation models
Particular Implementation model

Figure 6-5 Modelling Framework.

7.0 PARTICULAR ENTERPRISE MODELLING .

7.1 Definitions.

Activity Function. The Activity Function describes the actions/operations required to produce the defined Activity Outputs from the Activity Inputs provided. The Enterprise Activity Function is represented at the Intermediate Modelling Level as a set of Functional Operations.

Activity Input. Activity Input describes the information and/or material the Activity Function needs for its execution. The following inputs are defined, each with a specific purpose:

Primary Input: The primary Activity input consists of the objects to be actioned on by the Activity Function (Data and/or Material). Information and Material can serve as primary input because of their different logistic properties (e.g. information can be easily replicated at several sites whilst material cannot).

Secondary Input: The secondary Activity Input consists of the constraints on the action of the Activity Function.

Tertiary Input: The tertiary activity input is the means (resource) required to execute the action of the Activity Function .

Activity Output. Activity Output describe the information (or material) the function produces as a result of its execution. The following Activity Outputs are defined, each with a specific purpose:

Primary Output: The primary Activity Output consists of the objects resulting from the action of the Activity Function (Data and/or Material). Information and Material can serve as Primary Output because of their different logistic properties (e.g. information can be easily replicated at several sites whilst material cannot).

Secondary Output: the secondary activity output consists of the resulting status of the Activity Function.

Tertiary Output: The tertiary activity output is the status and means returned from the Activity Function.

Basic Information. Basic information is an Information class that supports many company departments. It contains the company standards and guide-lines, and the equipment and organization descriptions valid for the whole company.

Business Process. A Business Process is a construct which defines the desired sequence of Enterprise Activities or other lower level Business Processes to be executed in order to perform a desired task, which produces a defined result. Business Processes are triggered by Business Process Events and executed according to the flow of action expressed in its associated Procedural Rule Set and operate under the influence of external constraints (Declarative Rules). Business Process Results are generated at the completion or termination of a Business Process and describe its end product. See figure 7-1.

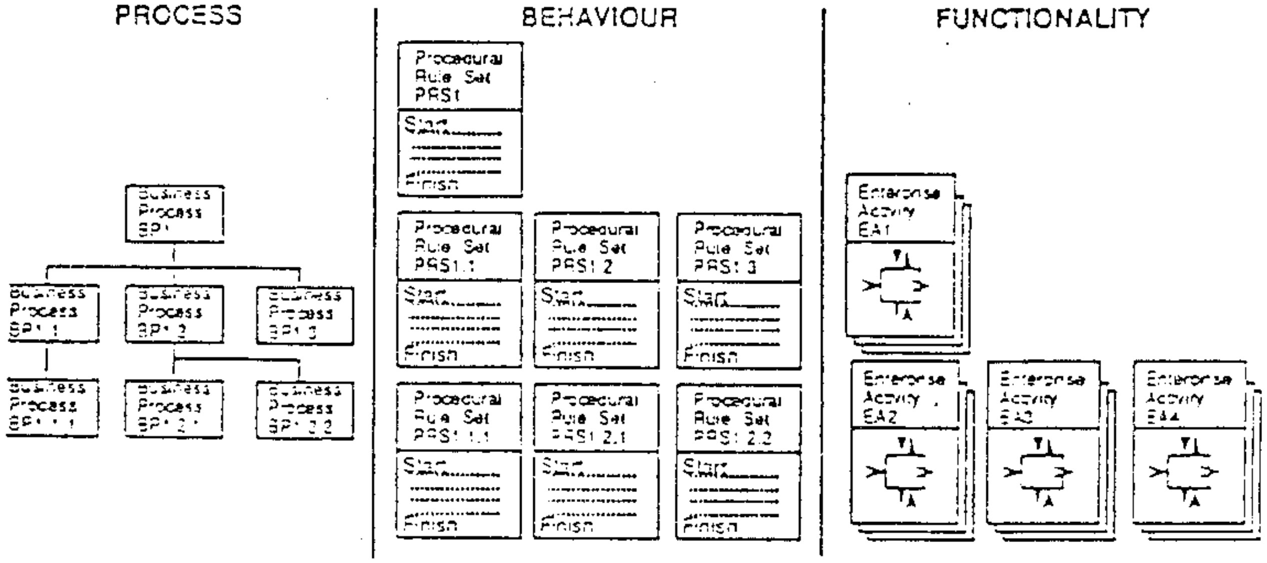


Figure 7-1 Business Processes.

Business Process Event. A Business Process Event is an event from outside of the Business Process which triggers the execution of the Business Process by initiating the processing of the associated Procedural Rule Set and thereby results in the activation of a group of Enterprise Activities which together comprise that Business Process.

Business Process Type: Business process Types are the basic, classified constructs at the generic architectural level from which Business Processes at the particular architectural level can be instantiated. Three Business Process Types are identified (see also Enterprise Activity Types):

Management oriented
Operational oriented
Support oriented

Component Catalogue. The component catalogue is a list of components which will be assembled in the particular enterprise, based upon components offered by vendors and/or particularly developed or available enterprise components. The component catalogue includes Information Technology and Manufacturing Technology components.

Conceptual Schema. The Conceptual Schema is one part of the ISO recommended "Three Schema Approach" and covers the central rationalised description of all enterprise information.

Declarative Rule. Declarative Rules define the long term enterprise constraints applicable to the execution of Business Processes and/or Enterprise Activities.

Enterprise activity. An enterprise activity is the lowest level description of a enterprise task to be performed. This description provides the operational parameters for the execution of business processes. Enterprise activities are separately described and are not part of any given business process, but can be employed by one or more business processes via their associated procedural rule sets. This relationship allows enterprise activities to be shared between different business processes and ensures a separation of functionallity (Enterprise Activities) and behaviour (Procedural Rule Set), making it possible to revise behaviour, in order to meet changing circumstances, without altering the installed functionallity. An Enterprise Activity is composed out of Activity Function, Activity Input and Activity Output see figure 7-2.

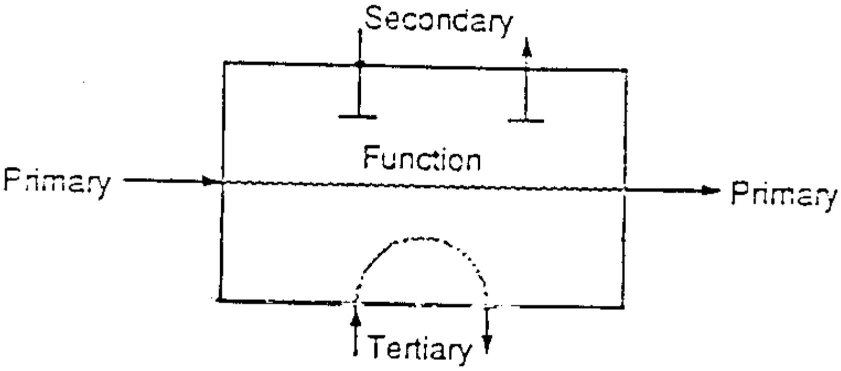


Figure 7-2 Enterprise Activity.

The inputs and outputs which are part of each enterprise activity form the basis of the External Schemata for that enterprise activity.

There is a strong relationship to the Functional Operations and the Functional Entities.

Enterprise Activity Types. Enterprise Activity Types are the basic, classified constructs at the generic architectural level from which Enterprise Activities at the particular architectural level can be instantiated. Enterprise Activity Types can be categorized according to their primary purpose (function) and can be further subdivided by categories according to the unique function of each enterprise activity.

Note: Three main categories of enterprise activities are identified until now, each associated with one of the main categories of business processes and each subdivided into Enterprise Activities and Functions.
Management oriented enterprise activities:
Plan
Control
Monitor
Report
Operational oriented enterprise activities:
Develop (Research, Define, Design, Qualify)
Produce (Rest, Move, Make, Verify)
Market (Search, Acquire, Sell, Distribute, Maintain)
Support oriented enterprise activities:
Install
Set-up
Maintain
Repair

Additional levels of subdivision may be required to obtain a powerful set of generic building blocks for the functional description of the enterprise.

Enterprise Domain. An Enterprise domain is the universe of discourse for particular modelling activities.

Event. See Business Process Event.

External Schema. The External Schema is one part of the ISO recommended "Three Schema Approach" and covers the way information is presented outside of the physical storage media.

Functional Entity. The Functional Entity is an implementation independent functional object able to send, receive, process (modify or interrogate) and optionally store information. Functional Entities have their own data that can only be accessed by their associated (virtual) processing facility. The data of a Functional Entity is generally referred to as its state, and the style of internal operation is called state oriented. However the internal functioning of a Functional Entity can be completely hidden from its partner. All capability is defined by the sets of Transaction Types provided at the

Functional Entity Input/Output Channels. The performance of a Functional Entity always results in a request to be sent to a partner entity or in a response to a partner entity (upon a request sent by this partner). Once a Functional Entity has received a request/response data unit from a partner it takes full responsibility of this data. The partner then is unable to re-access his issued data. Functional Entities will be defined at the Implementation Function View. Five Functional Entity Types are defined:

Data Storage Functional Entities,
Application Functional Entities,
Human Functional Entities,
Machine Functional Entities,
Communication Functional Entities.

Functional Operation. The Functional Operation is a construct of the Intermediate Function View and details the user oriented Enterprise Activity. It is the smallest unit of work that can be performed in the enterprise and ensures that Enterprise Activities can be executed in different ways, without changing the Enterprise Business Process. The Functional Operation will be executed by a Functional Entity in a 1:1 relation. Changes in the execution of the Enterprise Activity will be initiated by changing the Functional Operations and the Procedural Rule Set within that Activity. Five types of Functional Operations are identified:

Data Storage Functional Operations,
Application Functional Operations,
Human Functional Operations,
Machine Functional Operations,
Communication Functional Operations.

Internal Schema. The Internal Schema is one part of the ISO recommended "Three Schema Approach" and covers the structure of information in the physical storage.

Information Class. The Information Classes structure the enterprise information according to the user's views. Four Information Classes are defined, Product Information, Manufacturing Planning and Control Information, Shop Floor Information and Basic Information.

Note: Case study reports have identified additional information classes and efforts have to be made to rationalise any differences. The classes defined ultimately influence the physical storage structures.

Information Entity: A Information Entity is any concrete or abstract piece of information in the universe of discourse

Information Technology Components are required to process and distribute data for all activities in the enterprise. The Information Technology therefore includes the application programs, and the services that support the execution of application software by allowing them to run on the host hardware, while keeping application programs and hardware independent of each other. These services also support the Manufacturing Technology Components by enabling them to communicate with the host system and each other. Information Technology Components are:

Basic data processing resources include:-
services to execute application programs,
computer hardware,
basic software including microcode and firmware,
communications networks,
operating system software to control computer resources,
database management systems,
language compilers,
'housekeeping' and other supervisory software.

Application Software which is written for a particular application.

Implemented Functional Entity. An Implemented Functional Entity is a real existing Functional Entity which is released for use in the Operational Environment. It is not a model. Five types of Implemented Functional Entities are identified:

Implemented Data Storage Functional Entities,
Implemented Application Functional Entities,
Implemented Human Functional Entities,
Implemented Machine Functional Entities,
Implemented Communication Functional Entities.

Logical Cell is a building block for the definition of a group of logically structured resources required to support a set of related enterprise activities.

Manufacturing Planning and Control Information. This information contains all information necessary for handling orders. Therefore the class contains information like customer or production orders, logistic information concerning raw materials or tool devices, current capacity constraints, etc. The required information are accessed at the shop floor.

Manufacturing Technology Components are required to process materials, assemble products, pack and move them. Manufacturing Technology includes both people and their associated workplans and machines with their required control programs.

Many machines like robots and FMS contain 'intelligence' i.e. computer-like features. However, they are considered to be part of Manufacturing Technology since this intelligence is physically imbedded within the component.

People are referred to as part of Manufacturing Technology within the Enterprise Implementation Model, because they are an essential component of a CIM system and it is necessary to describe them in functional terms.

Metadata. Metadata is the data associated with a view and describes the structure and semantics of the data in the view.

Procedural Rule Set. The Procedural Rule Set represents the flow of control within enterprise Business Processes and/or Enterprise Activities. They define selection criteria for executing the lower level Business Processes, Enterprise Activities and/or Functional Operations. There is one and only one Procedural Rule Set per enterprise Business Process and Enterprise Activity. The Procedural Rule Set contains a series of statements, one for each lower level Business Process, Enterprise Activity or Functional Operation in the cluster, to define what action is required upon completion or termination of each Functional Operation, Enterprise Activity or lower level Business Process.

Product Information. Product information describes all features of a product and its production processes. It comprises behavioural, functional and physical aspects of the product such as the Bill of Material, the product geometry, production process description, process plans, CLDATA, etc.

Organizational Cell is a building block for grouping responsibilities within an enterprise. The Organizational Cell provide a basis from which decisions on the timely provision of resources and data needed for the execution of Enterprise Activities can be enabled and from which decisions can be made on the enterprise operation.

Shop Floor Information. Shop Floor information addresses the current operations in the production and assembly shops. Therefore, we find process oriented information such as shop schedules translated from production orders, the actual use of resources, and technological oriented information such as work procedure descriptions for machine tools.

Transaction. A Transaction is a single bidirectional exchange of messages from a requestor to its responding partner and from the responder to the requestor in that order. The Transaction includes three sequential steps:

1. requestor sends a request data unit to the responder.
2. responder acts upon the request.
3. responder sends a response data unit to the requestor.

7.2 Enterprise Modelling Level.

Prior to working on enterprise views the Enterprise Domain to be modelled will be defined, the external relations to other Enterprise Domains will be described, the events causing actions in this Enterprise Domain and the objects related to this Enterprise Domain will be identified.

7.2.1 Enterprise Function View.

In a first step Enterprise Function Views are generated by identifying the enterprise tasks. It may be necessary to describe a task by a series of sub-tasks. Enterprise tasks are then described using the concept of the Business Process, where it may be necessary to divide complex Enterprise Business Processes into subprocesses to produce the desired result of the higher level Business Process. For each Business Process the object to be acted on, the event which triggers the execution, the constraints, the expected result and the following Business Process(es) must be defined.

The second stage in building the Enterprise Function Model is to define the Procedural Rule Sets for the Enterprise Business Processes. During the generation of the Procedural Rule Sets the Enterprise Activities and their required capabilities necessary to establish the expected results of the Business Processes are defined..

The third stage is to define, in general terms, the Inputs, Outputs and Functions of each defined Enterprise Activity.

Figure 7-3 shows the three stages of the Enterprise Function View. The generation starts with the task decomposition into a hierarchical set of Business Processes. This decomposition leads to a formal definition of the Business Processes and their internal structure and sequence of actions (Procedural Rule Sets). The decomposition also leads to the definition of a set of Enterprise Activities and their constituent parts.

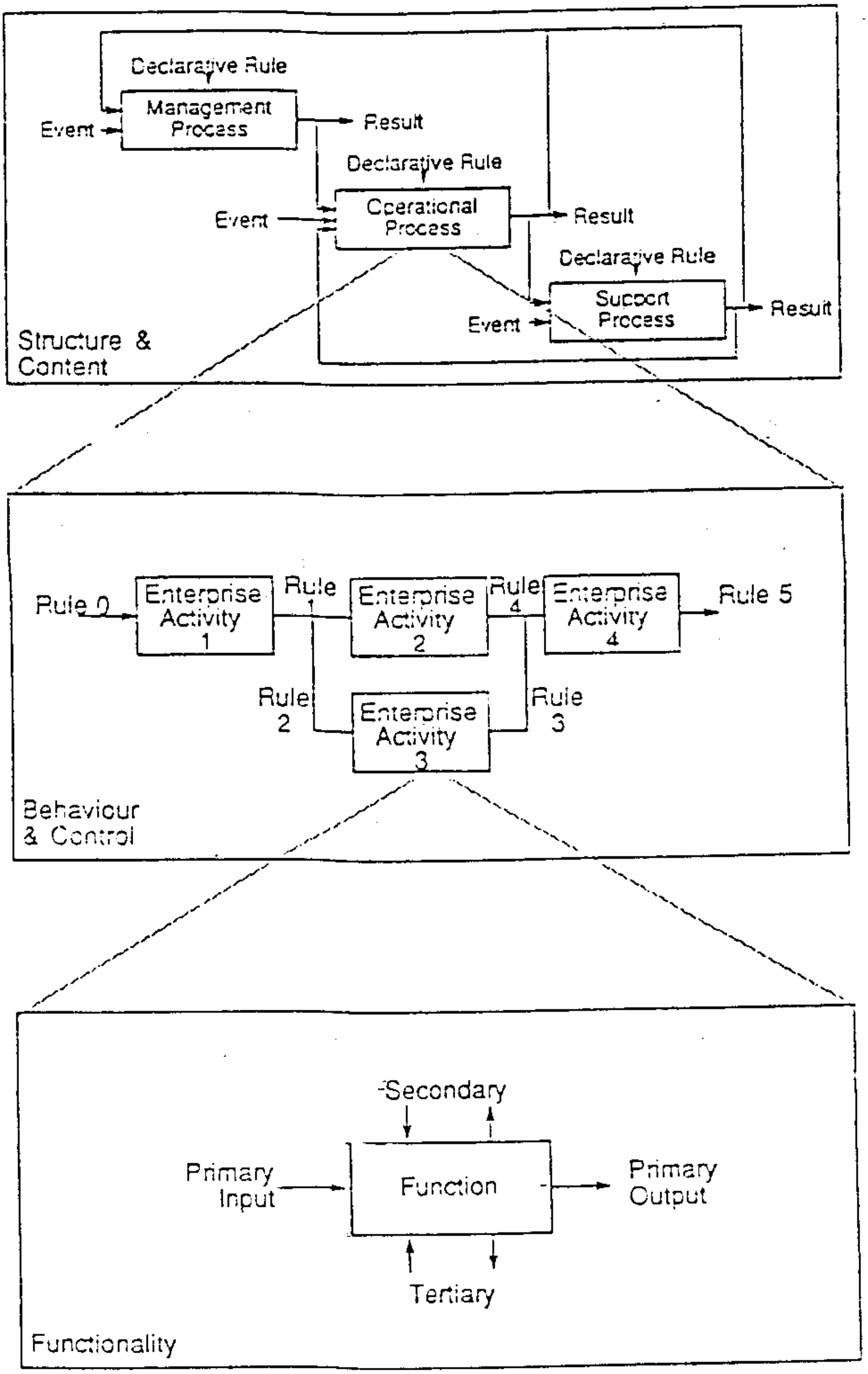


Figure 7-3 Business Process Building Blocks.

.2.2 Enterprise Information View.

The Enterprise Information View is required to capture the information needs of the particular enterprise activities defined in the Enterprise Function Model. Therefore it is necessary to generate the Information Inputs and Outputs of the Enterprise Activities defined. This generation can either be done for a selection of Enterprise Activities within a Business Process, or for all the Enterprise Activities after all Business Processes are defined.

The business user will select Information Classes and Information Items for each Enterprise Activity. Both, Information Classes and Information Items, will be provided from the Generic and Partial Architecture Level. Specific details have to be described as attributes to the Information Entities.

7.2.3 Enterprise Resource View.

During the definition of the business requirements the user has defined the required resources as tertiary inputs of the required Enterprise Activities. This Information is to be restructured in the Resource View to provide a consistent view of all the resources needed for a particular task of the enterprise. The Enterprise Resource Model is the basis for the Implementation Resource View which is required for further organisation of the resources in terms of physical location, and for identifying responsibilities.

7.2.4 Enterprise Organisation View.

Within enterprises the definition and identification of responsibilities is a important subject. Responsibilities have to be known by the CIM System for exception handling and human decision making processes. Responsibilities may be defined for enterprise assets (resources, information, capital, etc.) or for operational entities (Business Processes, Enterprise Activities, Functional Entities, etc.) The Enterprise Organization Model is the basis for the Implementation Organisation View which describes all the enterprise responsibilities.

7.3 Intermediate Modelling Level.

The final CIM System description (the Implementation Modelling Level) is derived from the business requirement definitions (the Enterprise Modelling Level). To isolate the two levels and to reduce the impact of changes from one level to the other the Intermediate Modelling Level is defined to act as a stable base between the business requirements definition and the implementation description. Business volumes will be applied to the defined business processes and activities as the volumes have a significant impact on the methods how a defined activity will be executed. The distribution of workload, the functional specifications of the executing entities and their physical location will be defined.

In this role the Intermediate Modelling Level represents the optimised user requirements, taking into account the expected volumes and all the enterprise business and CIM System constraints. This optimisation is carried out by organizers which optimize the different user requirements from a global enterprise view in terms of business needs and CIM System capabilities.

7.3.1 Intermediate Function View.

The Intermediate Function View is logically the realization of the Enterprise Function View. As such it links the relevant user requirement definitions (Business Processes and Enterprise Activities) to the relevant CIM System description at the Implementation Modelling level. For that purpose the tertiary input to the defined Enterprise Activities will be analysed and the Functional Operations necessary to generate the expected results will be defined. Next the Procedural Rule Set for each Enterprise Activity and its associated Functional Operations is defined. With the definition of the Functional Operations and the expected business volumes applied, specifications for the Human Functional Entities and the Machine Functional Entities, which will execute the described Functional Operations, will be generated.

7.3.2 Intermediate Information View.

The Information Items specified in the Enterprise Information Model will be structured into information objects, the necessary views for each object and if necessary the type of view editions which are expected during the life cycle of the object.

The Enterprise Information Model and the constructs and shells provided by the Intermediate Information view will assist the enterprise designer in generating the particular intermediate information Model. From these constructs and shells the enterprise designer will select available neutral schemas like SET, IGES or STEP for CAD applications or EDIFACT for commercial applications. Private conceptual schemas will be defined as necessary.

Based on the Information Items defined, the conceptual schemas selected and the data volumes expected, specifications for the Data Storage Functional Entities, Application Functional Entities and Communication Functional Entities will be defined. Specifications for the Application Functional Entities include the definition of the associated metadata.

7.3.3 Intermediate Resource View.

The Functional Operations defined at the Intermediate Functional View are identified independently from each other, probably causing redundancies. Existing redundancies will be identified and resolved during the generation of the Intermediate Resource View. Thus the Resource View at the Intermediate Modelling Level provides an optimised and balanced set of resources which have to be provided to satisfy the needs of the Enterprise Activities.

Within the Intermediate Resource View Logical Cells will be identified. The primary purpose of Logical Cells is to identify collections of equipment and resources which are candidates for having a high degree of integration because they support groups of functions which require close or frequent interaction. They may be used to reflect a job-oriented structure, or a process-oriented one and can be used for the generation of a physical layout.

There is some relation to the Organisational Cells, described in the Intermediate Organisation View. This relation is described in the Intermediate Organisation View.

7.3.4 Intermediate Organisation View.

The Organization View at the Intermediate Modelling Level describe an optimised and balanced organization of enterprise responsibilities for enterprise assets (resources and information) and for its operational entities (business processes, activities, objects, etc.). The Intermediate Organisation View includes responsibilities for "out of line situations" such as breakdown of equipment and associated recovery, or update of information items in the data base in case of problems. The responsibilities have to be organized in order to satisfy the needs of the enterprise for decision making. There is relation between the Logical Cells of the Intermediate Resource View and the Organisational Cells of the Intermediate Organisation View.

Relationship between Resource and Organization View. Figure 7-4 illustrates how the two Views are providing their contribution for the structuring of resources. The logical structure of the resources consider the equipment according to its use by Enterprise Activities. Logical Cells collect all resource objects (human beings, machines, data storage capabilities, data processing capabilities, etc.) required for implementation of one or a group of Enterprise Activities. On the other hand Organisational Cells structure the enterprise resources according to their provision or their identified responsibilities. Both Views provide relevant information for the arrangement of resources by the Resource Management at the Implementation Architecture Level of the Particular Enterprise Model.

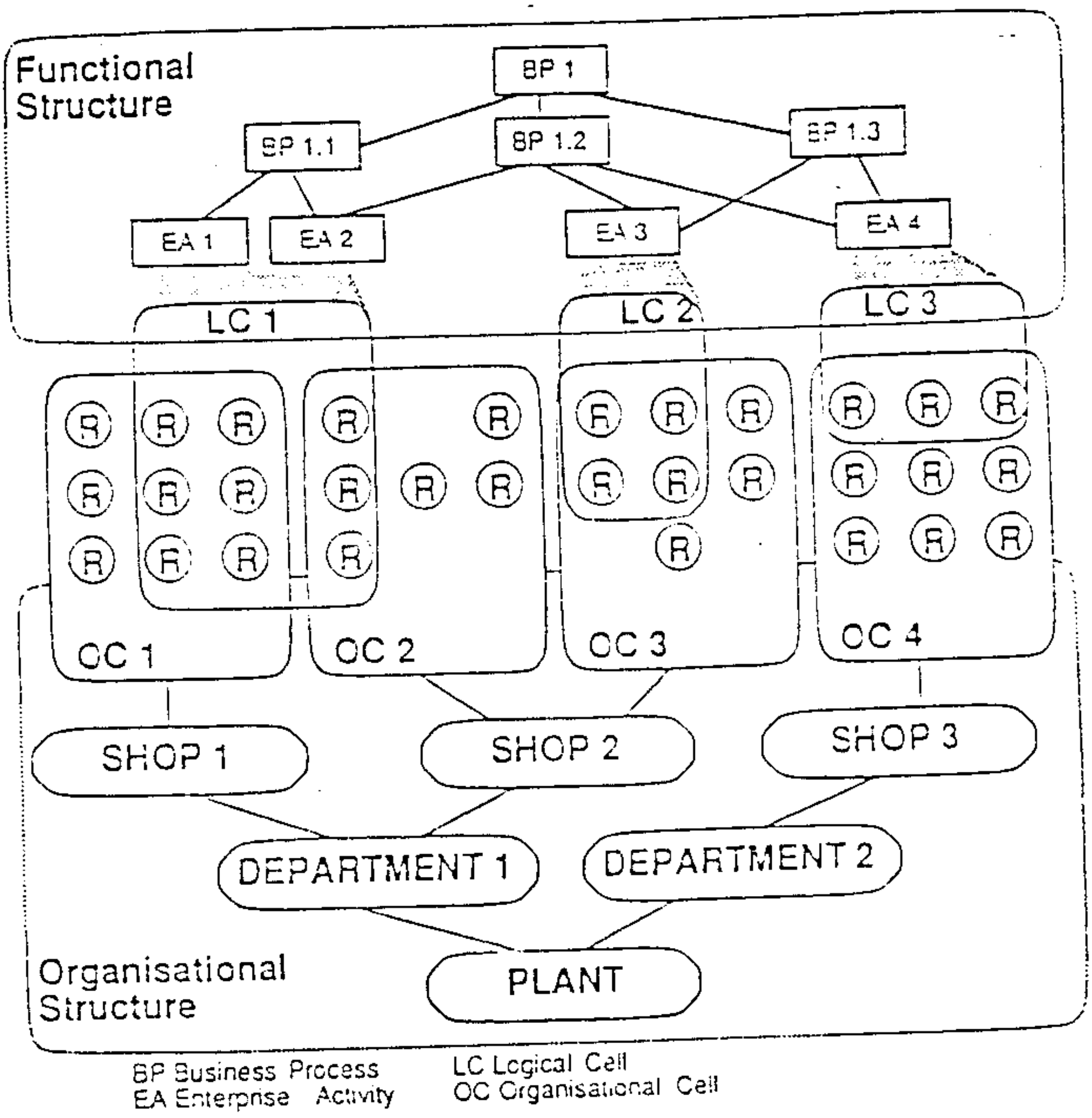


Figure 7-4 Relationship resource versus organisation.

Such relationships exist for other identified responsibilities in the Organization View and their related

Enterprise Objects in the Information and Function View as well.

7.4 Implementation Modelling Level

The Implementation Modelling Level defines finally and in detail the Information Technology and Manufacturing Technology components for all Business Processes and Activities defined at the Enterprise Modelling Level and all Functional Operations and Functional Entities defined at the Intermediate Modelling Level. The Information Technology and Manufacturing Technology components will be selected and ordered from the components catalogue. After delivery the components will be tested and after successful testing the model of the component will be transferred into the released Enterprise Implementation Model and the actual component will be installed in the Enterprise Operational Environment, where it will used for the actual execution of the enterprise tasks. The components installed in the Operational Environment are referred to as Implemented Functional Entities.

At Implementation Modelling Level two different models are defined. The Implementation Model and the Released Implementation Model. The purpose of keeping both models apart is to use the Implementation Model for test purposes without any impact to the Operational Environment and its model, when changes are required after the Operational Environment has been put in action.

7.4.1 Implementation Function View.

Based upon the specifications of the Functional Entities, generated at the Intermediate Function Level, and the component catalogue the Functional Entities to be implemented will be selected and ordered or manufactured. After availability of the Functional Entities they will be verified against their specification and in cooperation with the already implemented components. After successful verification the Functional Entity will be released for operation; that means the model of the Implemented Functional Entity will be transferred to the released implementation model and the actual entity will be placed into the operational environment. After successful testing of the Business Processes and Enterprise Activities together with their Procedural Rule Sets and in conjunction with the associated Functional Entities they will be transferred into the operational environment, where they can be instantiated, whenever associated events occur.

7.4.2 Implementation Information View.

Based upon the specifications of the Data Storage Functional Entities, generated at the Intermediate Information Level, and the component catalogue the Data Storage Functional Entities to be implemented will be selected and ordered. After availability they will be verified against their specification and the data necessary for the operational environment will be loaded. The Application Functional Entities as defined at Intermediate Information level will be selected and ordered from the component catalogue and applications not available from the catalogue will be designed. Both, the loaded data and the applications will, after successful verification against the specification, be released to the operational environment; that means the model of the Implemented Data Storage Entities and the specification of the Implemented Application Entities with their associated metadata will be transferred to the released implementation model and the implemented entities will be transferred to the operational environment.

7.4.3 Implementation Resource View.

Based upon the Intermediate Resource Model including the Logical Cells, the Implemented Entities of the Implemented Function Model and the Implemented Information Model, the Implementation Resource Model will be built and released.

7.4.4 Implementation Organisation View.

Based upon the Intermediate Organisation Model including the Organisational Cells and the Implementation Resource Model the organisational structure and the responsibilities for the Implemented Entities and Data will be defined.

7.5 Summary.

Figure 7-5 shows, for a particular enterprise, the process of stepwise derivation through all views, as well as the different user inputs at the three Modelling Levels.

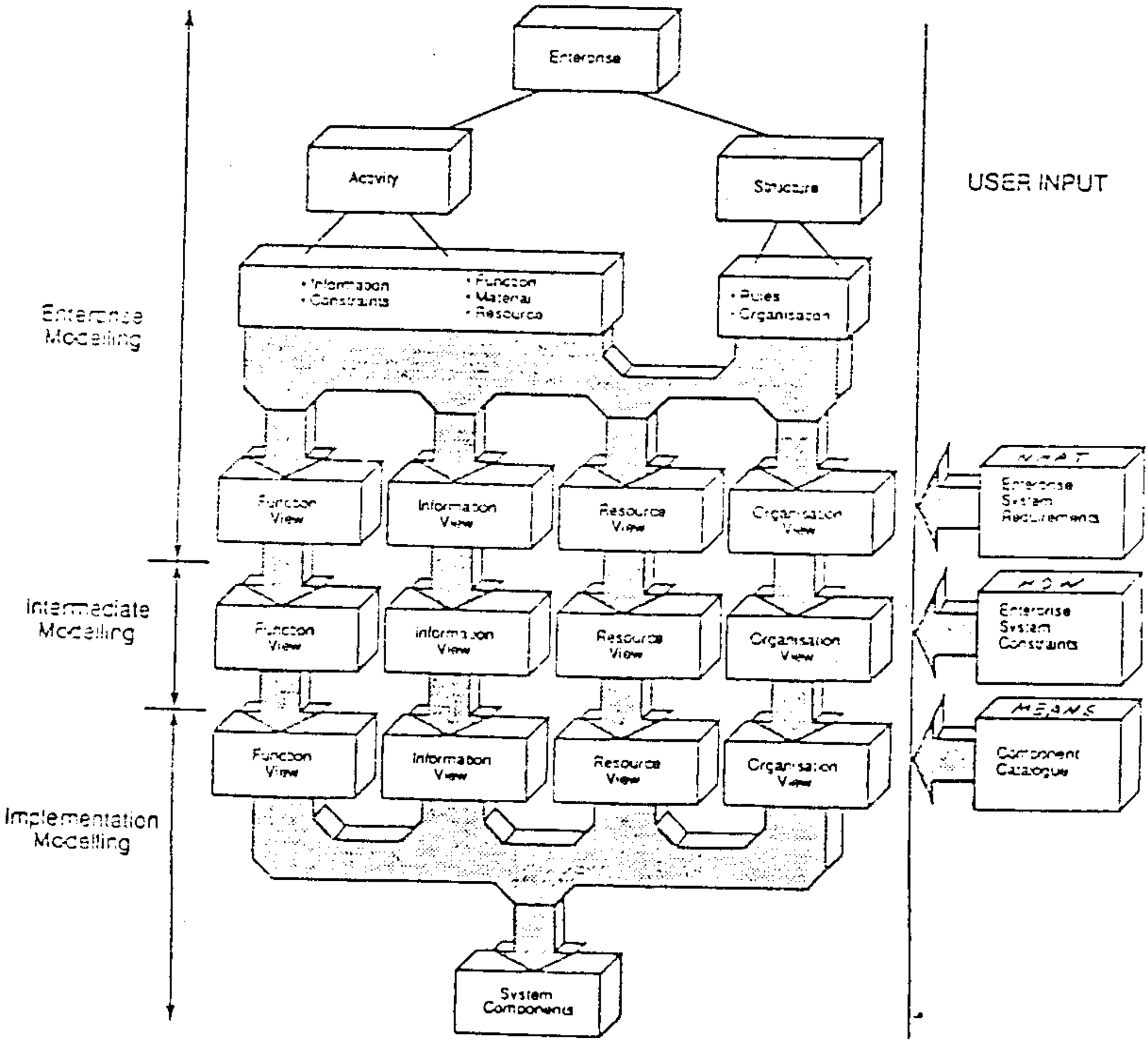


Figure 7-5 Enterprise Modelling.

8.0 IMPACT ON STANDARDISATION.

After this framework has been accepted to be the base standard for enterprise modelling in the CIM environment, the concepts described in clauses six and seven will lead to identification of business domains, business processes and enterprise activities which are common for many enterprises within the same or in different industries.

To ensure, that the designers of enterprise models do not define the same business processes, enterprise activities, Functional Operations, and Functional Entities again and again and also to ensure that Enterprise Engineering Functions support the enterprise designer with already defined types, standardisation of Business Process types, Enterprise Activity types, Functional Operation types and Functional Entity types is one of the most urgent activities. Also, within an Enterprise Reference Model Domains can be identified, including its tasks and boundaries to other Domains.

For these defined types the required data can be identified, the semantics and necessary syntaxes can be defined and standardised.

> Note: A good example for such data identification is EDIFACT (Electronic Data Interchange for Administration, Commerce and Trade) which defines the data, syntax and standard messages for defined business activities. Another example is the emerging standard STEP which defines data for a set of other enterprise activities.

Such identification and standardisation of data structures can, based on analysis of business processes, be defined for many domains. This will ease data exchange between different applications as well as it will avoid that the same data used in different applications will be defined and stored several times in different structures.

Formal, computer processable description techniques for Business Processes will enable that business processes are portable between enterprises using different Information Technology environments and will enable that knowledge based systems support the derivation from Enterprise to Implementation Modelling level.

Definitions of Functional Entities at the Shop Floor, which may be a result of the current activities in TC184SC5WG1 on a Shop Floor Reference Model will enable high level languages to be defined for the shop floor and may ease the attempts to define a global language for shop floor entities.

Definition of Functional Entities and their capabilities in the shop floor domain, will enable definition of tooltypes for these entities which, associated with appropriate parametric data stored in a library, can ease selection of tools and simulation of machine operation.

Agreement on the functionallity of the servers described in Appendix A of this document will enable standard protocols between the servers to be defined, thus ensuring communication between the servers in a heterogeneous Information Technology systems environment.

APPENDIX A: CIM SUPPORT SYSTEMS.

A.1 Definitions.

Basic Data Processing Resources are Information Technology services and devices on which the Enterprise Engineering Functions and the Integrated Infra Structure are implemented. This includes hardware as well as software.

Enterprise Engineering Functions are set of tools to support the Enterprise Modelling Process. The tools guide the enterprise designer through the steps to be performed, offer the constructs and shells for assistance, assist in describing the processes, activities and rules and making use of the shells, identify incompleteness or inconsistencies in the models, offer simulation tools and support the release to operation environment process.

Integrated Infra Structure a set of Information Technology services which are deemed necessary for the generation and execution of CIM systems complying to the CIM Reference Model described in this document.

A.2 The Integrated Processing Environment.

The total CIM Support System which supports the enterprise modelling process and the operational process is regarded as the Integrated Processing Environment which is split into two parts see figure A-1 :

1. the enterprise engineering environment,
2. the enterprise operational environment.

The Enterprise Engineering Environment guides the business user in the enterprise modelling process and consists of:

a. the Enterprise Engineering Functions,
b. the Integrating Infrastructure,
c. the Basic Data Processing Resources.

The Enterprise Operational Environment guides the execution of the enterprise tasks and consists of:

a. the Released Enterprise Implementation Model,
b. the Integrating Infrastructure,
c. the Basic Data Processing Resources,
d. the Implemented Functional Entities.

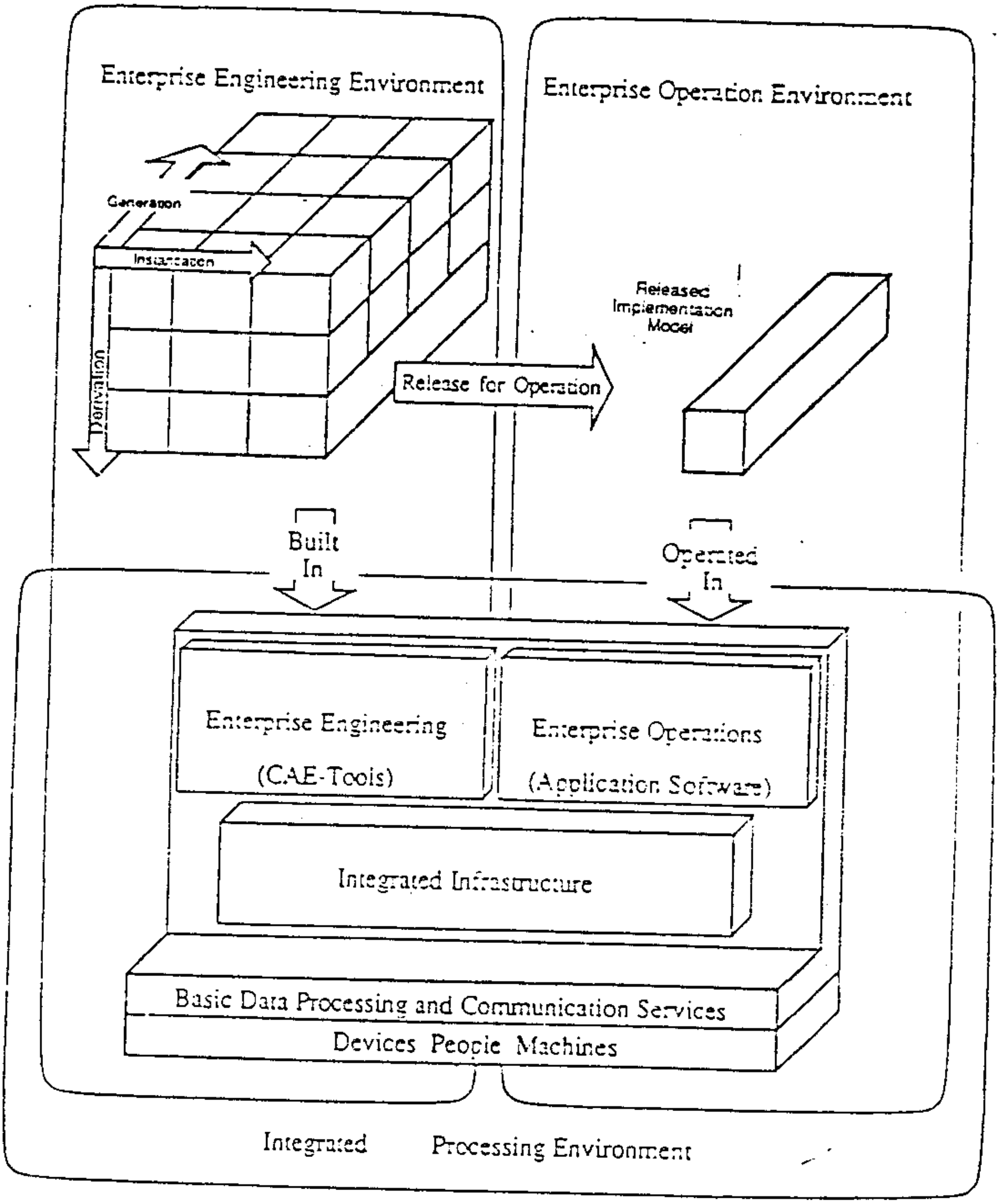

Enterprise Engineering Environment
Enterprise Operation Environment
Generation
Instantiation
Derivation
Released Implementation Model
Release for Operation
Built In
Operated In
Enterprise Engineering
(CAE-Tools)
Enterprise Operations
(Application Software)
Integrated Infrastructure
Basic Data Processing and Communication Services
Devices People Machines
Integrated Processing Environment

Figure A-1 Integrated Processing Environment.

A.3 The Integrating Infrastructure

The Integrating Infrastructure (IIS) is divided into the following Services (figure A-2) :

Front End Related Services,
- Human Front End,
- Machine Front end,
- Application Front End,
- Data Front End,
- Communication Front End,

Business Related Services,
- Activity Control,
- Resource Management,
- Business Process Control,

Information Related Services,
- System Wide Information Services,

Intrasystem Communication Services,
- System Wide Exchange Service.

All Services are Functional Entities with a defined functionallity. They communicate with each other via protocols and make use of the client-server concept. Two types of protocols are defined the access protocol and the agent protocol. The access protocols are used for communication between different services, regardless whether these services are within the same physical system or in distributed systems. The agent protocols are used for communication between distributed parts of the same service. The different services are explained briefly in the following subclauses. A detailed specification will be provided at a later time in a separate document.

A.3.01 The Human Front End Service.

The purpose of the Human Front End Service is to provide a consistent, implementation and application independent service to the Implemented Human Functional Entities at CIM System built time and CIM System run time. The service enables the humans to interact with the CIM system in a homogeneous way independent of the connected application programs and it enables the CIM system to communicate with the implemented Human Functional Entities via different interaction devices homogeneously.

A.3.02 The Machine Front End Service.

The main purpose of the Machine Front End Service is to represent external data processing equipment such as Numeric Controllers, Robot Controllers, Programmable Controllers, etc. within a CIM system in a homogeneous way. The service allows to monitor and control the Implemented Machine

Functional Operations (IMO) by way of IMO controllers. The IMO Controller is a configurable element that allows the Machine Front End designer to complement each particular functional object (e.g. a machine program) in such a way that it behaves as the Implemented Machine Functional Operation. The set of IMO-Controllers constitute the configurable Application Interface for the interaction with remote machines.

A.3.03 The Application Front End.

The purpose of the Application Front End Service is to provide a defined set of services for the interaction between the Implemented Application Functional Entities and the services of the Integrating Infra Structure.

A.3.04 The Data Management Front End

The purpose of the Data Management Front End Service is to support the Implemented Data Storage Functional Entities by providing local data storage and retrieval to be used in association with System Wide Data, and enabling the insertion into CIM systems of DBMSs or other means of data storage in a standardized fashion.

A.3.05 The Communication Front End.

The Communication Front End provides a means for the IIS Services to access transparently the OSI Environment or the private Communication Environment and enables the transfer of information using real communication system.

A.3.06 The Activity Control Service.

The purpose of the Activity Control Service is to provide a defined set of services to control, monitor and service the (possibly distributed) Enterprise Activities. The service provides status information on Enterprise Activities, manages the execution of Enterprise Activities and reports out of line situations from the executing Front Ends.

A.3.07 The Business Control Service

The purpose of the Business Process Control Service is to provide the services to control the (possibly distributed) Business Processes. The service manages the execution of Business Processes, controls the sequencing and synchronization of Enterprise Activities by invoking the rules, and allows for comprehensive CIM System-user visibility and control over Business Processes.

A.3.08 The Resource Management Service.

Resource Management Service has the purpose of overall, system-wide, management of the enterprise resources as required for the proper execution of Business Process occurrences and their constituent Enterprise Activities. Such management may be achieved in either a centralized or distributed manner (according to user choice), by assigning responsibility for given (sub)sets of resources to specified Resource Management Services.

A.3.09 The System Wide Data Service.

The purpose of the System Wide Data Service is to provide the locating of the data, the consistency of data and the management of schema conversions The service allows access to data only to duly authorized requestors and maintains system wide data and information integrity according to systems rule,

A.3.10 The System Wide Exchange Service.

The purpose of the System Wide Exchange Service is to provide the adequate service needed to transfer access-, agent-, and external- protocols occurring among data oriented services, process control oriented services and front end services, and enables its service users to remain unconcerned with the notion of nodes and hence with any networking issue and to know their correspondents only as IIS Services Entities.

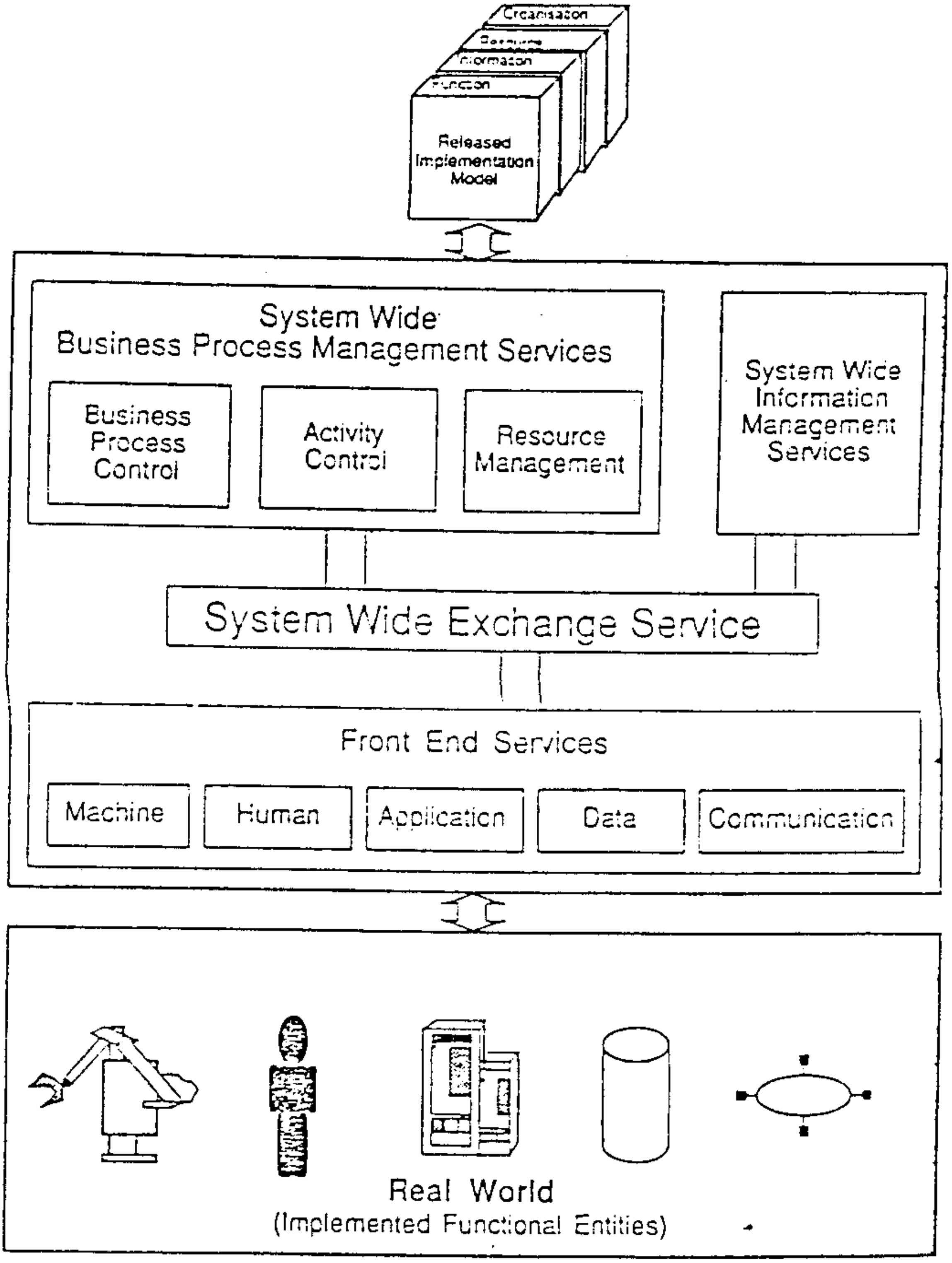

Organisation
Resource
Information
Function
Released Implementation Model
System Wide Business Process Management Services
Business Process Control
Activity Control
Resource Management
System Wide Information Management Services
System Wide Exchange Service
Front End Services
Machine
Human
Application
Data
Communication
Real World
(Implemented Functional Entities)

Figure A-2 Integrating Infrastructure.

(부록2)

Reference model for shop floor production standards

ISO/TC 184 N 147

TECHNICAL REPORT 10314 Part 1

Reference Model for Shop Floor Production Standards

Part 1
A Reference Model for Standardisation and a Methodology for Identification of Standards Requirements

Foreword

This report was created by ISO TC184/SC5/WG1 to help speed the timely development of standards in the field of industrial automation. The work is being published in the form of a technical report because it is not possible, in view of the current state of the art of modelling for manufacturing, to draw up an international standard which would be complete and precise and which would not be too restrictive in this rapidly changing field. The report is intended as a guideline and will be reviewed and augmented periodically.

Contents

0 Introduction

1 Scope and Field of Application
- 1.1 Scope
- 1.2 Field of Application

2 Terminology
- 2.1 General
- 2.2 Specific Items

3 Abbreviations

4 Objectives for Manufacturing Standardisation
- 4.1 General information
- 4.2 The Standards Viewpoints

5 The Reference Model for Shop Floor Production
- 5.1 General Information
- 5.2 The Context of Shop Floor Production
- 5.3 The Shop Floor Production Model (SFPM)
- 5.4 The Generic Activity Model (GAM)

6 Methodology for extracting areas of standards
- 6.1 General Information
- 6.2 Proposed Approach
- 6.3 Matrix Representation of Identification Procedures

7 Summary

0 Introduction

This report is intended to provide a tool to help identify and co-ordinate present and future activities involving ISO and IEC standards work in the field of industrial automation. Specifically, the Reference Model developed within the technical report is to be applied to the area of Discrete Parts Manufacturing. For the purposes of this report the word "manufacturing" should hereafter be interpreted to mean Discrete Parts Manufacturing. While the Reference Model may have application beyond the area of Discrete Parts Manufacturing, the developers of the Reference Model have not tested the Reference Model in other areas of industrial automation activities.

Since the model and the methodology will need to be refined so as to adjust to emerging technologies, a technical report rather than an international standard has been chosen as the means of presentation.

This technical report does not include the development of individual standards themselves, but rather the establishment of a common framework, in terms of a Reference Model, to assist future standards development.

The Reference Model for standardisation must be:

- simply structured, flexible, modular and generic,

- based upon readily available and acceptable terminology,

- able to be applied to a wide range of manufacturing operations and organisations, recognising the need to interface equipment and systems to human beings,

- independent of any given, predetermined realisations in terms of system configurations or implementations,

- open-ended in its ability to be extended, and in its ability to encompass new technologies without unreasonably invalidating current realisations

- independent of existing technologies in manufacturing automation and computer science.

This Reference Model and its methodology are to be used to identify areas for standardisation and will benefit organisations involved in developing such manufacturing standards. It will also be of interest to the manufacturing community consisting of both suppliers and users, but it is not intended to be a design for system integration of manufacturing.

The report for the Reference Model for Shop Floor Production Standards is comprised of two parts. Part 1 describes the Reference Model and methodology for identification of possible standards requirements. It addresses the following issues:

a) a review of existing models and modelling methodologies.

b) the derivation of an initial, generic, standards classification scheme for manufacturing.

c) the adoption of a functional view of a manufacturing enterprise.

d) the establishment of an initial reference model according to the results of a) - c).

e) the development of a methodology for extracting areas of standards.

Part 2 describes the application of this Reference Model and methodology for extracting areas of standards. It addresses:

i) the application of the methodology in order to derive a particular list of areas for required standards.

ii) the identification of areas for standards.

iii) the derivation of standards requirements.

The separate development of Part 2 may show that modifications are necessary to the highly interrelated Part 1. Normal ISO procedures will address this issue.

This document is Part 1 and consists of this introduction and seven chapters. In Chapters 1, 2, and 3 respectively, the scope and the field of application, description of terms and abbreviations are described. The objectives of standardisation for manufacturing are described in Chapter 4. Chapter 5 contains the Reference Model. Chapter 6 introduces the methodology for applying the Reference Model to clarify and extract areas of standards and this methodology is further amplified in Part 2. Chapter 7 provides an overall summary of the document.

1 Scope and Field of Application

1.1 Scope

This report presents and describes a means of identifying where new or revised manufacturing standards may be required. It establishes a Reference Model for Shop Floor Production, which is then used as the basis for developing a methodology for the identification and extraction of areas for standards. The assumptions used to develop the Reference Model are:

- the field of interest is the manufacture of discrete parts and in particular the production (physical realisation) of these parts.

- the Reference Model needs to be open-ended so that it can be revised to incorporate new technologies, and

- the Reference Model needs to be generic in nature so that it can be applied to a wide range of applications and is not directed to a particular organisational structure of manufacturing.

It is emphasised that the Reference Model:

- provides a conceptual framework for understanding manufacturing and

- can be used to identify areas of standards necessary to integrate manufacturing systems.

The Reference Model does not however provide a methodology for designing, implementing, operating and maintaining any existing or future manufacturing automation system. There may be a need to develop other Reference Models which can be used for those purposes, perhaps based on the work described in this report. The development of such models is beyond the scope of this technical report.

1.2 Field of Application

The Reference Model described in this report is intended for use in the identification of standards within the Shop Floor Production area of manufacturing.

Manufacturing is perceived to be all inclusive, from customer order through to delivery of the product. Twelve manufacturing functions have been identified as being a part of manufacturing. The following is a list of these twelve functions, together with illustrative, non-exhaustive activities typically related to these functions:

1) Corporate management, e.g.
   - Direction of enterprise
   - Strategic planning
   - Feasibility study for investment
   - Risk management

2) Finance, e.g.
   - Financial planning
   - Corporate budgeting
   - Financial accounting

3) Marketing and sales, e.g.
   - Marketing research
   - Advertising
   - Sales forecasting
   - Sales scheduling
   - Pricing
   - Sales (order, delivery, invoice)
   - Product service

4) Research and Development, e.g.
   - R & D planning
   - Basic research
   - Applied research
   - Product development
   - Manufacturing development

5) Product design and Production engineering, e.g.
- Define product specifications
- Preliminary design and testing
- Detailed design
- Design analysis, test, evaluation
- Revise design
- Release design for production planning
- Project management
- Process planning
- Programming of numerical control and programmable control
- Tooling
- Plant engineering
- Bill of material
- Quality assurance planning of production
- Production configuration

6) Production management, e.g.
- Production scheduling
- Product and Inventory control
- Production monitoring
- General maintenance request
- Quality control
- Cost control and cost management

7) Procurement, e.g.
- Vendor performance
- Purchasing
- Receiving
- General stores

8) Shipping, e.g.
- Product storage
- Distribution

9) Waste material treatment, e.g.
- Waste material processing
- Waste material storage

10) Resource management, e.g.
- Facility management
- Tool control
- Energy management
- Time and Attendance
- Facility security
- Health and Safety
- Environment control

11) Maintenance management, e.g.
- Preventive maintenance
- Corrective maintenance

12) Shop Floor Production, e.g.
- Material store
- Transport material
- Transform material
- Incoming inspection
- In-process gauging and testing
- In-process audit
- Product audit

As shown in Figure 1 and described in the definitions which follow in Section 2.2, these functions tend to be grouped under three main headings. Items 1) through to 4) are functions of the Enterprise concerned with strategic long term planning activities. Items 5) through to 11) are functions of the Facility, concerned with tactical planning of the production process, resource management and product modelling. The final item, 12) Shop Floor Production, is a function which involves the activities that actually create a physical product.

The twelve manufacturing functions are interrelated and a single Reference Model covering all twelve functions would be desirable. After careful study of existing work, it was decided that the development of a single Reference Model covering every function of manufacturing was not manageable at this time.

The area of Shop Floor Production on the other hand has shown an urgent need for and a willingness to adopt standards. The Reference Model described in this technical report is intended to guide the planning for and the development of standards to assist the integration of an automated Shop Floor Production system. It is recognised that the Shop Floor Production function will be required to interface with functions (and their activities) outside the scope of Shop Floor Production itself. Figure 2 is a clarification of how major functions of manufacturing might be interrelated.

In the future, Reference Models for manufacturing that include Enterprise and Facility functions may be developed. Any future modelling work in the area of manufacturing should take account of the Reference Model for Shop Floor Production presented here and every effort should be made to ensure compatibility between the Reference Model for Shop Floor Production and any Facility or Enterprise Reference Models that may be developed.

2 Terminology

2.1 General

A number of terms are described in this chapter to provide a better understanding by the user of this report. These descriptions are intended to be used solely in the context of this report and are not intended to be general definitions.

2.2 Specific terms

2.2.1 Reference Model: a means of describing the activities and components of manufacturing through the use of figure(s) and text.

2.2.2 Discrete Parts Manufacturing: systems of functions for producing products or parts consisting of discrete elements.

2.2.3 Function: a grouping of several activities performed to realise some manufacturing objective.

2.2.4 Activity: a manufacturing process which causes some change in inputs.

2.2.5 Level: a collection of activities which form a degree of subordination in a hierarchical arrangement.

2.2.6 Enterprise: an entire manufacturing unit consisting of a corporate component and one or more Facility components. The corporate component is responsible for interactions between the external environment of the Enterprise and the Facility or Facilities, and also for the control of functions within the Facility or Facilities.

2.2.7 Facility: a component of an Enterprise which excludes corporate functions. The Facility is responsible for providing support and direction for Enterprise and Shop Floor Production activities.

2.2.8 Shop Floor Production: a component of a Facility whose function is directly related to the production of discrete parts and/ or products.

2.2.9 Shop Floor Production Model: the basic model used to describe the structure within Shop Floor Production.

2.2.10 Interaction: an interrelationship or interconnection between the Subjects within Shop Floor Production, and also between the Subjects and Actions within or external to Shop Floor Production.

2.2.11 Generic Activity Model: a generic model used to describe the execution of activities within Shop Floor Production and their interactions with functions interfacing to Shop Floor Production.

3 Abbreviations

Several abbreviations are used in this report:

• GAM	Generic Activity Model
• SFPM	Shop Floor Production Model
• ST	Store
• TF	Transform
• TP	Transport
• VE	Verify

4 Objectives for Manufacturing Standardisation

4.1 General information

4.1.1 The objectives of standardisation

ISO has pointed out the objectives of standardisation as follows:

- mutual understanding
- health, safety, protection of environment
- interface, interchangeability
- fitness for purpose
- variety control.

4.1.2 The concept of Standards Viewpoints

The process of standardisation for manufacturing is to define areas of standards, to select aspects to be standardised and to define standards based on the state of the art. In this report, the following Standards Viewpoints (defined in 4.2) have been selected to identify the needs of standards in the manufacturing field derived from the guidance of the above ISO objectives:

- Safety
- Environment
- Compatibility
- Performance
- Operability
- Maintainability
- Reliability
- Qualifications
- Description

These nine Standards Viewpoints, together with the Reference Model, are used in the proposed methodology for extracting areas of standards.

4.2 Standards Viewpoints for Manufacturing

Nine Standards Viewpoints for manufacturing are defined as follows and are to be applied to any area where standards may apply:

4.2.1 Safety Viewpoint

Safety is concerned with the effect on safety that normal and erroneous operation of the manufacturing facility would have on the operating personnel, the equipment and the work in progress. It also covers the need for traceability, that is the ability to trace the sequence of manufacturing processes and components used in manufacture. Hence define design and operation standards to ensure safe operation.

4.2.2 Environment Viewpoint

Environment is concerned with the effect that operating entities can have on the physical environment both during the manufacturing activity itself and as a by-product of that activity. In each case both normal and failure modes should be considered. Hence identify standards which define each effect, how it is to be measured and norms for acceptable operation.

4.2.3 Compatibility Viewpoint

Compatibility is concerned with interchangeability and interdependence, and addresses issues related to interfacing. Hence define design and operation standards to ensure compatibility.

4.2.4 Performance Viewpoint

Performance is concerned with the performance aspects which can be categorised in terms of speed, quality of finish (output), resources consumed etc. Hence identify standards to define these qualities and appropriate specifications.

4.2.5 Operability Viewpoint

Operability is concerned with all aspects of human interactions with the manufacturing environment. It is essentially concerned with easy operation and the avoidance of improper operation under both normal and erroneous conditions. Hence define design and operation standards to ensure easy and correct operation.

4.2.6 Maintainability Viewpoint

Maintainability is concerned with reducing downtime and the risks and costs of evolution. It is very important to enable easy maintenance of all manufacturing constituents and the system itself, to maintain a known status for each manufacturing entity and its documentation and to support traceability where required. Hence define design and operation standards to ensure easy maintenance.

4.2.7 Reliability Viewpoint

Reliability is concerned with all aspects of how a system is designed and operated to meet specified levels of availability. This starts with the reliability of individual elements and components, and goes up to the reliability of the entire system and organisation. Hence define design and operation standards to ensure reliability.

4.2.8 Qualifications Viewpoint

Qualifications are concerned with the quality of personnel for achieving the proper design and operation of manufacturing entities. The viewpoint is essentially concerned with the training of operators as qualified experts. Hence define design and operation standards for the qualifications of personnel for managing manufacturing activities.

4.2.9 Description Viewpoint

Description is concerned with all aspects of how the design and operation of manufacturing entities are defined and described to provide all persons concerned with a means of mutual understanding. It is essentially concerned with the terminology, identification and description of documents. Hence define design and operation standards to ensure mutual understanding in the manufacturing field.

5 The Reference Model for Shop Floor Production

5.1 General Information

This chapter describes the Reference Model developed to be used in the identification of standards within Shop Floor Production.

The Reference Model presented in the following three sections is in many ways a synthesis of a variety of existing models. Firstly the major activities undertaken for manufacturing are identified as a context for Shop Floor Production. Secondly a Shop Floor Production Model (SFPM) is presented which groups Shop Floor activities into hierarchical levels. Thirdly a Generic Activity Model (GAM) is presented to model the various activities within each level of the SFPM.

Each activity in Shop Floor Production can be represented by an instance of the GAM. The various elements of the GAM will have different interpretations for any given activity.

The intent of this Reference Model is to assist in identifying where standards may be required and the detail is not sufficient for other purposes such as providing a basis for designing or implementing a manufacturing system. The Reference Model presents a functional view of the Enterprise, Facility and Shop Floor Production.

5.2 The Context of Shop Floor Production

Section 1.2 sets out the scope of manufacturing, identifying 12 major functions. Figure 1 shows one example of how Shop Floor Production takes place in a context established by those other functions. Accordingly interactions occur between many of the activities of Shop Floor Production and those of the Facility, and to a lesser extent with those of the Enterprise itself. The methodology described later allows for these interactions to be identified, in addition to the interactions occurring within Shop Floor Production itself.

5.3 The Shop Floor Production Model (SFPM)

Shop Floor Production is distinguished from the other eleven manufacturing functions identified in Section 1.2 by the fact that it contains those activities which are directly engaged in producing parts. It is common practice to group these activities into several levels.

In this report, a four level model is selected to present the activities of the Shop Floor Production function. It is quite likely that specific manufacturing implementations may require more or less than four levels, but four levels seem sufficient for the purpose of identifying areas of standards.

Also this report identifies four types of manufacturing activity, each type corresponding to each level of Shop Floor Production. There can be (and generally are) several activities in progress at the same time at each level. The types of activity are:

1) execute shop floor production processes
2) command shop floor production processes
3) co-ordinate shop floor production processes
4) supervise shop floor production processes.

The four level model in Figure 3 illustrates the structure of Shop Floor Production. This model is called the Shop Floor Production Model (SFPM). The figure shows the name of each level and the corresponding type of activities at this level together with its description.

5.4 The Generic Activity Model (GAM)

This technical report approaches the modelling of those characteristics relevant to standards identification within the SFPM through the concept of an activity. A Generic Activity Model (GAM) as shown in Figure 4 has been developed to model the execution of the various activities at each level.

The internals of the GAM represent an interrelated set of four Subjects and four Actions. The Subjects and Actions are defined below:

5.4.1 Four Subjects

a) **Control Information** which includes:

(i) command information, normally flowing from a higher level to a lower level, which initiates, alters or terminates an activity,

(ii) status information which is generated in direct response to a command and normally flows in the opposite direction to the command,

(iii) request information which corresponds to control information for control interaction within a level (peer to peer), if any, and

(iv) response information which corresponds to status information for control interaction within a level (peer to peer), if any.

b) **Data:** all information other than control information required for or resulting from the performance of an activity.

c) **Material:** Material is a production object of the the manufacturing activity. Material includes all the physical matter that enters the product during manufacturing: raw materials, parts and assemblies, auxiliary materials, products and scrap material.

d) **Resources:** Resources are all the physical means required to carry out the manufacturing that are not Material. Resources include transform, transport, verify and storage equipment; tools and fixtures; data processing and communication systems; basic resources such as supply material, energy, space and time; personnel.

Note: The general class of "**Information**" is defined here to consist of control information and data, that is as the union of a) and b) above.

5.4.2 Four Actions

a) **Transform:** The act of changing control information, data, material or resources from one form to another form, or one state to another state. Transform includes encoding or parsing information, decomposing commands, and cutting, forming, assembling, or adjusting material or resources.

b) **Transport**: The act of moving control information, data, material, or resources from one point in the enterprise to another.

c) **Verify**: The act of assessing the compliance of all transformed control information, data, material and resources to determine their conformance to a specification.

d) **Store**: The act of retaining control information, data, material or resources at a specified location within Shop Floor Production or Facility until they are required to be transported.

The GAM describes the execution of activities in terms of four Subjects (Control Information, Data, Material, Resources) and four Actions (Transform, Transport, Verify, Store). The GAM is developed from a view that the realisation of manufacturing activities can be represented in terms of these four Subjects and four Actions.

The representation of a given activity will be a specific instance (an instantiation) of the GAM in which specific correspondences (bindings) are established between the elements of the GAM (Subjects, Actions) and the entities which correspond to those elements for that particular manufacturing activity.

Not all elements of the GAM will be present in every instance - the lowest level of Shop Floor Production for example does not have command and status information for the non-existent level below.

Major manufacturing activities may be decomposed into (sub)activities in a horizontal fashion (into a series of further activities) or in a vertical/ hierarchical fashion (into levels of activities). The GAM is generic in the sehse that it can be applied to all activities.

Establishing specific GAM bindings to particular manufacturing activities yields specific Subjects and Actions. The A- and B-type procedures described in Section 6 can then make use of these specific Subjects and Actions in identifying areas of standards.

6 Methodology for extracting areas of standards

6.1 General Information

There are a number of interactions among activities, and among Subjects and Actions within activities, which are used to identify the area of standardisation in manufacturing. Therefore the methodology consists of examining the various activities, Subjects and Actions in various relationships and connections to determine if a standards requirement is defined.

The goal of standardisation is to permit more effective relationships and connections between two or more objects. In manufacturing, objects may be machine-to-machine relationships and/ or connections, or man-to-machine relationships and/ or interactions.

The Reference Model described in the preceding chapter is essentially declarative rather than procedural — that is it presents a set of concepts and a hierarchical approach by which a manufacturing Enterprise can be better understood. The Model defines useful manufacturing functions, characteristics and relationships amongst them. It is also necessary to show how they can be used to identify standards areas for manufacturing.

This chapter provides such a methodology for extracting areas of standards.

6.2 Overall approach

The way in which the various elements of the Reference Model are used to identify standards is illustrated in Figure 5. The various activities of manufacturing are grouped into twelve major functions. One of these, Shop Floor Production, is further modelled by a four level model, SFPM (Figure 3).

The various activities at each level of SFPM can each be modelled by the GAM (Figure 4), with each such instantiated model resulting in bindings for GAM concepts (Subjects, Actions). The activities of SFPM can be combined into vertical and horizontal arrangements and this fact is recognised in the procedures of 6.3 which identify potential areas for standards, each of which can be represented by a cell entry in the matrices of Figure 6.

Once a potential standards area has been identified (a non-empty cell entry) it should be examined in the light of

- the technologies of interest, i.e. the technologies that might be evident in the Subjects and Actions for that area, and

- the Standards Viewpoints described in Section 4.2.

These two concerns act as filters which restrict areas of standards to those which are realisable with current technology and which reflect the objectives of standardisation.

6.3 Extraction Procedures

These procedures use the concepts of the Reference Model and describe how these can be examined to identify potential areas for standards. The approach places strong emphasis on interactions between elements of a system. Consequently it is expected to produce areas of potential standards which are prerequisites for systems integration.

The various combinations in the Reference Model alluded to above are examined by applying the two kinds of procedures defined below to each of the Standards Viewpoints.

It is important to note that:

- each of the two kinds of procedure can be applied independently of one another,

- the same standard might arise from applying different procedures and

- the application of these procedures is not guaranteed to produce an exhaustive list of standards but rather to clarify areas of standards.

6.3.1 Procedures A: the "Interrelationship within a level" procedures

These procedures are used for extracting areas of standards for interrelationship between Subject and Action, Subject and Subject, and Action and Action corresponding to each level of the SFPM.

For each activity, these procedures should be applied to extract areas where standards may apply.

Procedure A1: **Subject-Action** interrelationship

For each activity in each level of the SFPM, consider any possible Subject-Action interrelationship for applicable areas of standards.

Procedure A2: **Subject** interrelationship

For each activity in each level of the SFPM, consider any possible Subject-Subject interrelationship for applicable areas of standards.

Procedure A3: **Action** interrelationship

For each activity in each level of the SFPM, consider any possible Action-Action interrelationship for applicable areas of standards.

6.3.2 Procedures B: the "External" procedures

These procedures are used for extracting areas of standards for interrelationship between activities for both vertical and horizontal structure.

For each activity these procedures should be applied to extract areas where standards may apply.

Procedure B1: **Horizontal** interrelationship

For each level of the SFPM consider any possible interrelationships between Shop Floor Production and other manufacturing functions (the context functions 1..11 of 1.2) for applicable areas of standards.

Procedure B2: **Vertical** interrelationship

For each level of the SFPM consider any possible Subject interrelationship with the level above and below for applicable areas of standards, and hence relevant Subject attributes.

6.4 Matrix Representation of Identification Procedures

Corresponding to each procedure (A1, A2, A3, B1, B2) it is possible to envisage a matrix which represents that procedure graphically. These are shown schematically in Figure 6. Each cell of each matrix represents a potential area for standards. Similar matrices are used in Part 2 of this report to show the standards areas identified through the application of these procedures, as well as worksheets for other users of the Reference Model and its methodology.

7 Summary

This document is Part 1 of a report on a means to identify where industrial automation standards for Shop Floor Production may be required. It has described a Reference Model which may be used as a systematic method for determining areas for standards development work. Part 2 of the report describes an application of the methodology which utilises this Reference Model in identifying these areas, catalogues them and cross-references them with standards already completed or in the process of development.

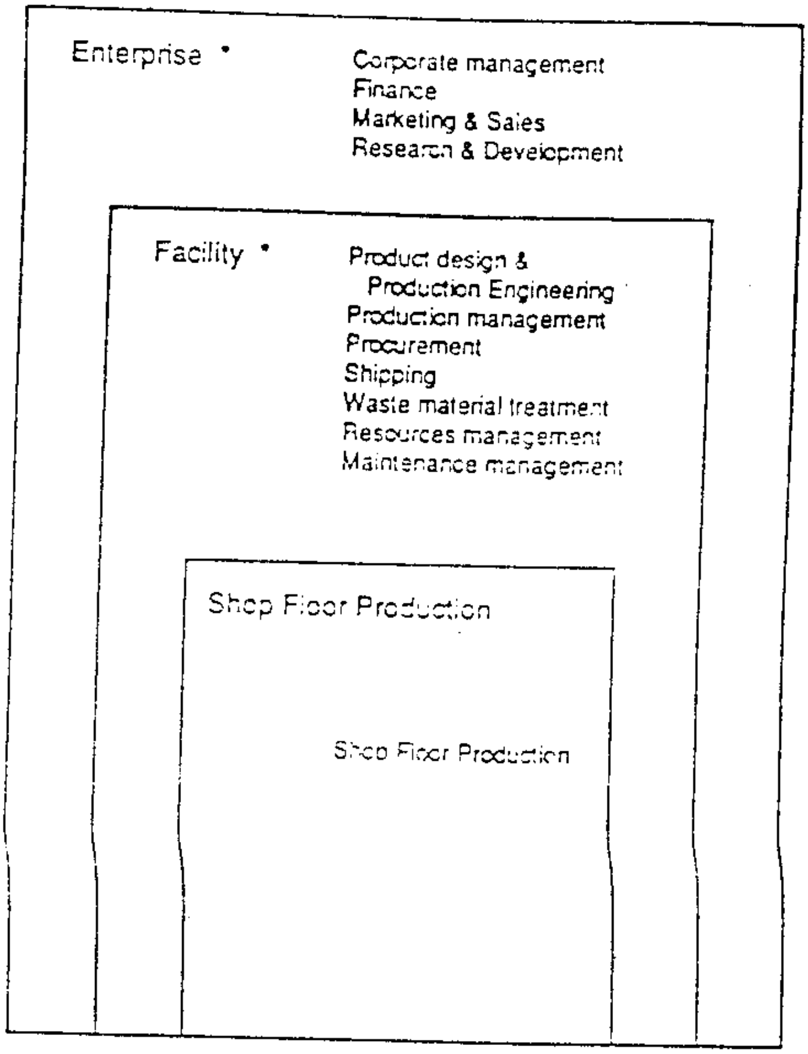


Figure 1 - Typical Grouping of Manufacturing

- * The combination of Enterprise and Facility form the context of Shop Floor Production as described in 5.2.

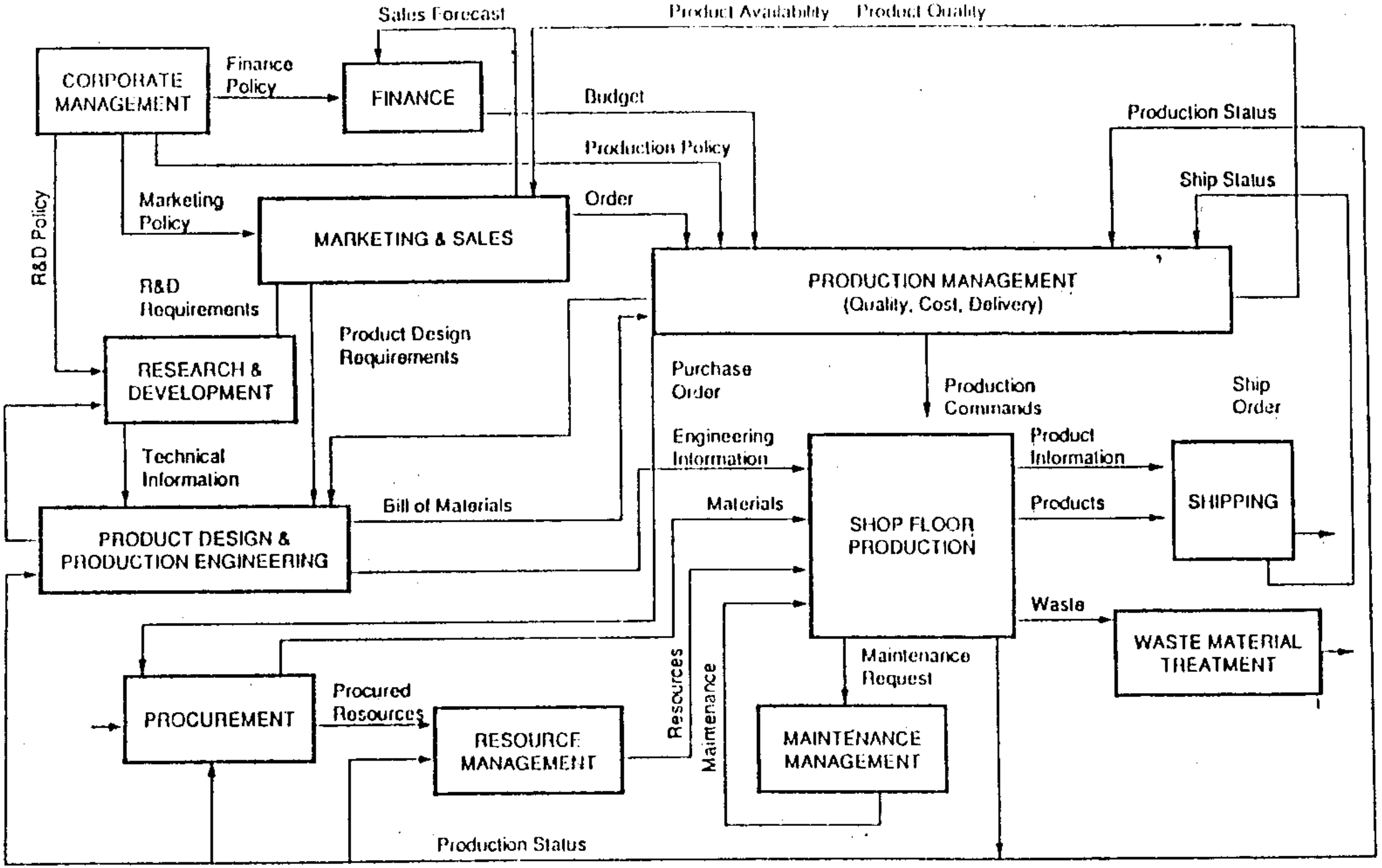
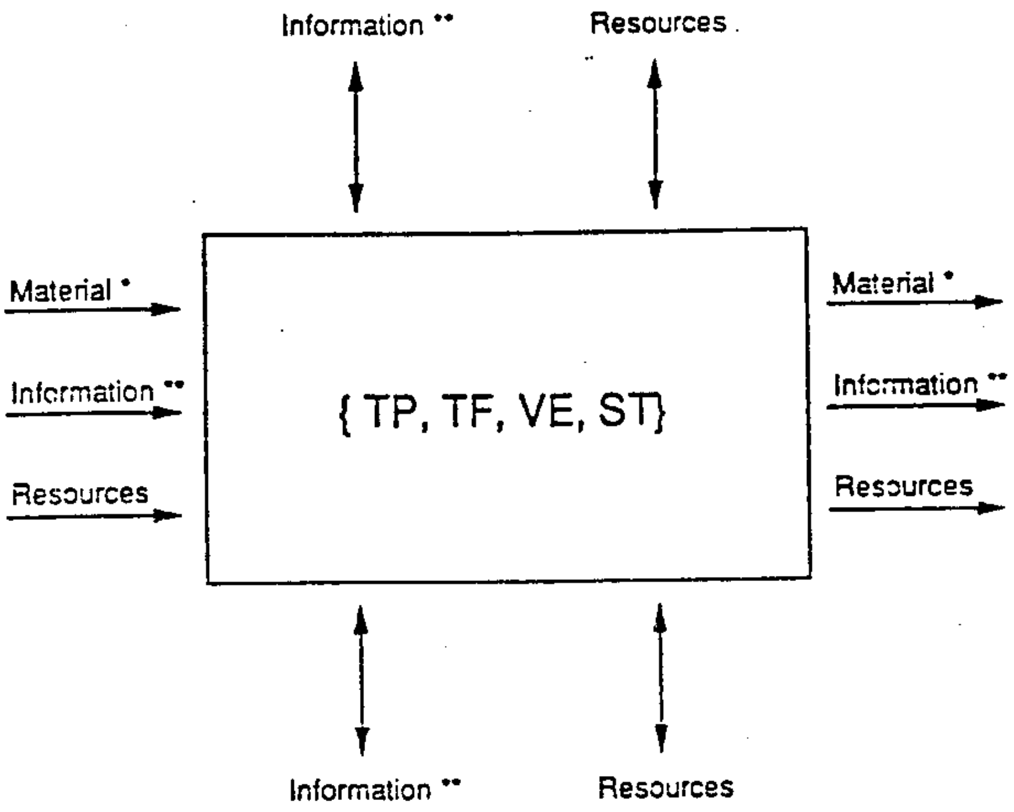


Figure 2 - Typical Arrangements of Manufacturing Functions

	Level	Sub-Activity	Responsibility
4	Section /Area	Supervise shop floor production process	Supervising and co-ordinating the production and supporting the jobs and obtaining and allocating resources to the jobs
3	Cell	Co-ordinate shop floor production process	Sequencing and supervising the jobs at the shop floor production process
2	Station	Command shop floor production process	Directing and co-ordinating the shop floor production process
1	Equipment	Execute shop floor production process	Executing the job of shop floor production according to commands

Figure 3 - Shop floor production model (SFPM)



* Actions {TP, TF, VE, ST} on Material are defined only at Level 1

TP	= Transport
TF	= Transform
VE	= Verify
ST	= Store

** Information is defined in the text to include both control and data components. For strict hierarchies, horizontal Information flows are restricted to data components

Figure 4 - Generic Activity Model

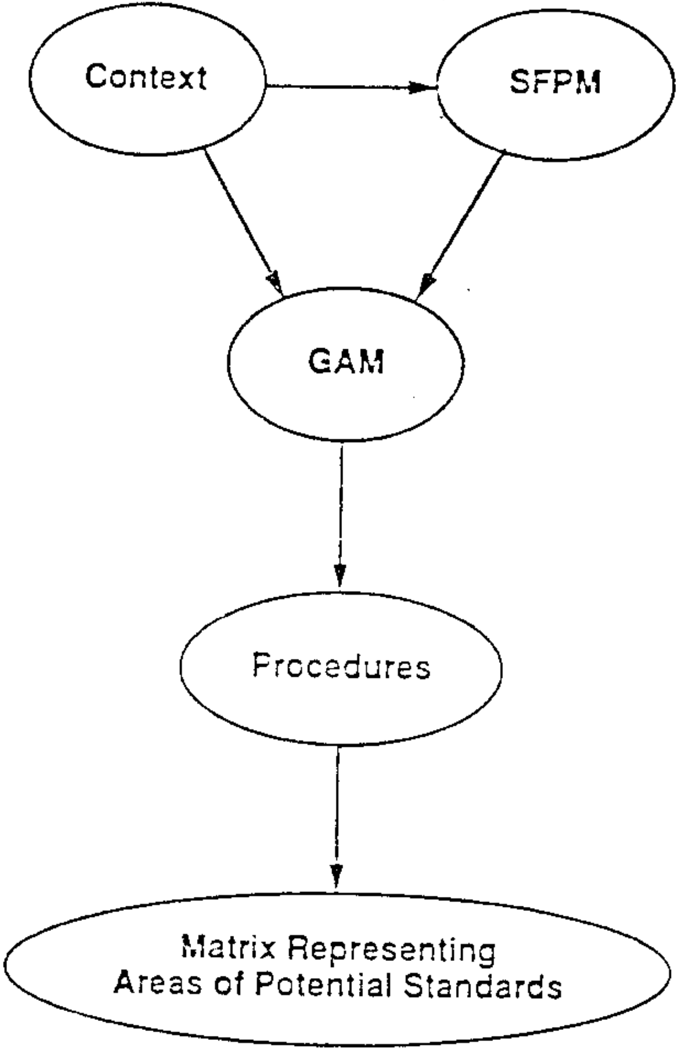


Figure 5 - Overall Approach to identifying areas of standards

NOTE: Part 2 will also use concepts of appropriate technologies and Standards Viewpoints to restrict potential areas for standards into those which are practical and desirable.

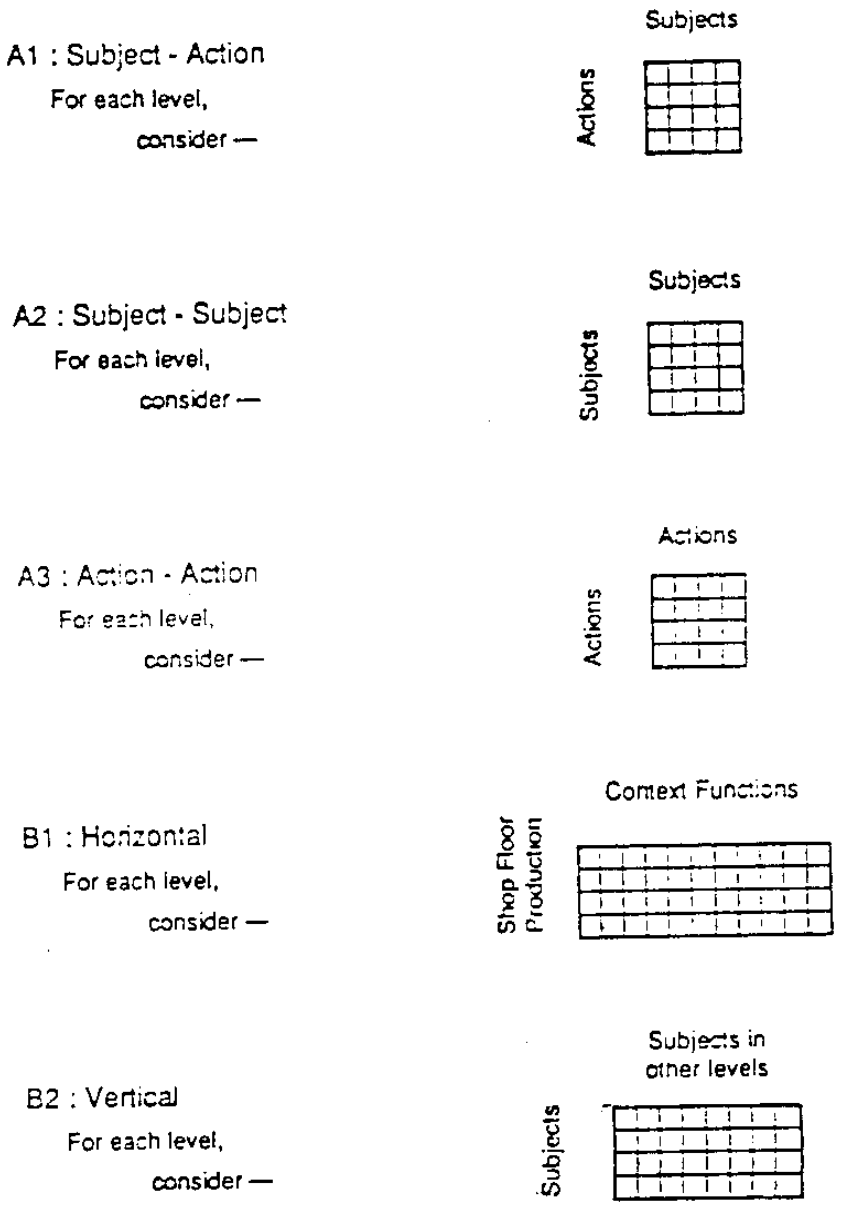


Figure 6 - Matrix representation of identification procedures

(부록3)

Status of PDES - related activities

U.S. DEPARTMENT OF COMMERCE
National Institute of Standards and Technology

NISTIR 4432

National PDES Testbed
Report Series

Status of PDES - Related Activities

(Standards & Testing)

Cita Furlani
Joan Wellington
Sharon Kemmerer

U.S. DEPARTMENT OF
COMMERCE
Robert A. Mosbacher,
Secretary of Commerce

National Institute of
Standards and Technology
John W. Lyons, Director

October 1990

Table of Contents

I. Introduction 1

II. Standards Organizations Developing STEP 1
- ISO Committee TC184/SC4 1
- IGES/PDES Organization 3
- ANSI U.S. Technical Advisory Group 5

III. Standards Activities Related to STEP or PDES 5
- NIST-Wide Product Data Exchange Task Group 5
- Digital Representation of Product Data Standards Harmonization Organization . . 6
- Navy/Industry Digital Data Exchange Standards Committee (NIDDESC) 6
- ASME Standards Development Committee Y14 6
- Mathematical Definitions of Dimensions and Tolerances 7
- Apparel Product Data Exchange Specification 7
- Application Interface Specification 7
- Dimensional Measurement Interface Specification 7

IV. Testing Activities 7
- National PDES Testbed 8
- PDES, Inc. 8
- CALS Test Network 9
- IGES Testing 9
- Federal Information Processing Standards 9

VI. International Programs 10
- Commission of the European Communities 10
- Conformance Testing Services (Phase) Two 10
- European Strategic Programme for Research and Development in Information Technology (ESPRIT) 11
- STEP/ENV (Draft European Standard) 11
- Advanced Project for European Information Exchange (APEX) 12
- Japanese STEP Translator Development Project 12
- Computer-Aided Manufacturing - International (CAM-I) 12
- A European Computer Integrated Manufacturing Architecture (AMICE) 13

VII. Summary 13

Acknowledgements 13

References 14

Additional Reading 16

Appendix 17

STATUS OF PDES-RELATED ACTIVITIES
(Standards & Testing)

Cita Furlani
Joan Wellington
Sharon Kemmerer

I. Introduction

This report describes the standards and testing activities that relate to PDES (Product Data Exchange using STEP). PDES is the name given to the United States organizational activity that supports the development and implementation of the international Standard for the Exchange of Product Model Data, informally known as STEP. The definition of PDES was made more explicit in March 1990 by altering the meaning of the acronym from its earlier one, "Product Data Exchange Specification."

Much of the information in this paper was originally collected as part of the justification for a national initiative to implement automated product data sharing. Other papers describing product data sharing in various industrial sectors were also produced and the set will be published in late 1990.

II. Standards Organizations Developing STEP

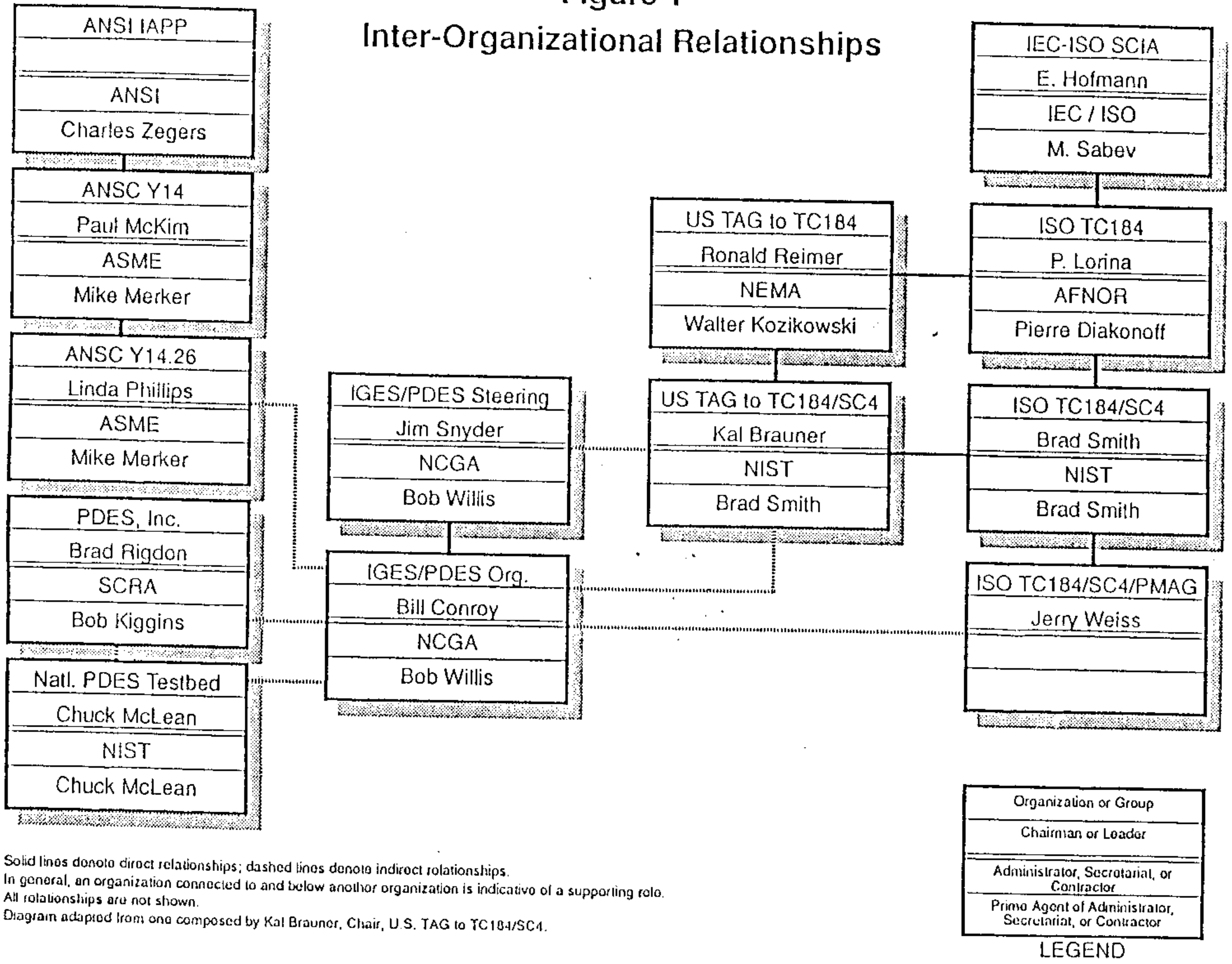
A number of organizations are working both nationally and internationally to develop an exchange specification for product data, known as STEP. This section describes the efforts of three of these organizations and their relationships [Figure 1]:

- ISO Committee TC184/SC4,
- IGES/PDES Organization, and
- ANSI U.S. Technical Advisory Group.

ISO Committee TC184/SC4 - In 1983, the International Organization for Standardization (ISO) reached unanimous agreement on the need to create a single international standard which would represent a computerized product model in a neutral form throughout the life cycle of the product without loss of completeness and/or integrity.

In December of that year, the ISO initiated Technical Committee 184 (TC184) on Industrial Automation Systems and formed Subcommittee 4 (SC4) to work in the area of representation and exchange of digital product data [1].

Figure 1
Inter-Organizational Relationships



Solid lines denote direct relationships; dashed lines denote indirect relationships.
In general, an organization connected to and below another organization is indicative of a supporting role.
All relationships are not shown.
Diagram adapted from one composed by Kal Brauner, Chair, U.S. TAG to TC184/SC4.

Today 26 countries are involved in the work of SC4, 18 are participating members (P-members), and 8 are recognized as Observers. The United States is a P-member, and as such, has one vote on issues before SC4. The SC4 Secretariat is currently held by NIST [2]. Technical support for SC4 comes predominantly from Working Groups (WG2-7).

IGES/PDES Organization (IPO) - Within the United States, technical activities to support the development of STEP have been ongoing since 1985. The U.S. organization which has led these activities is the IGES/PDES Organization [3]. The IPO is composed of over 550 volunteers from government, industry, and academia and is chaired by a representative from the National Institute of Standards and Technology (NIST).

The concept for the development of a standard such as STEP grew out of an earlier effort by the IPO known as IGES (Initial Graphics Exchange Specification). The goal of IGES was to develop a neutral data format which would allow the digital exchange of information among computer-aided design (CAD) systems.

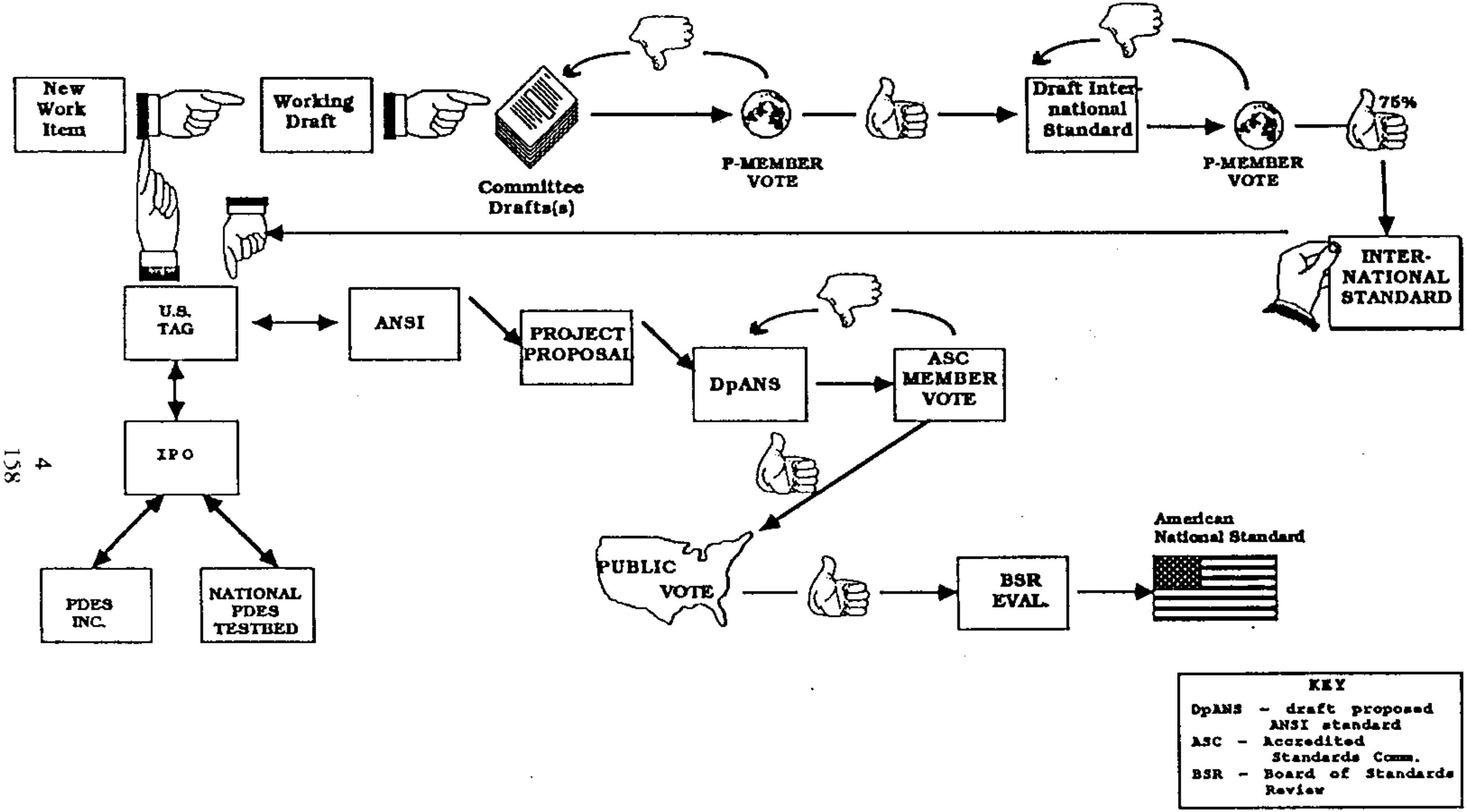
The first version of IGES was published in 1980 and was adopted as ANSI (American National Standards Institute) Standard Y14.26M in 1981. Additional versions of IGES were issued in 1983, 1986, and 1988. The last two were adopted as ASME/ANSI standard Y14.26M-1987 and ASME/ANSI standard Y14.26M-1989, respectively. IGES Version 5.0 was released in September 1990 [4].

With IGES, the user can exchange product data models in the form of two- and three-dimensional wireframe representations as well as surface representations. Translators convert a vendor's proprietary internal database format into the neutral IGES format and from the IGES format into another vendor's internal database. The IGES effort has not focused on specifying a standardized information model for product data. IGES technology assumes that a person is available on the receiving end to interpret the meaning of the product model data.

Members of the IPO, drawing on experience gained with IGES, recognized that a more sophisticated standard would be required to support the integration of different types of product life cycle applications. In 1985 a formal study, called the PDES Initiation Effort, was begun which established a framework and methodologies for the subsequent PDES activities. The PDES effort is focusing on developing a complete model of product information that is sufficiently rich to support advanced, state-of-the-art applications [5].

Approximately 250 technical representatives from the United States and other countries meet four times a year for a week at a time to address IGES/PDES/STEP-related technical issues. Twice a year, these IPO Quarterly Meetings are held concurrently with ISO Committee TC 184/SC4 working group meetings. Many of the technical participants from the U.S. and other countries are active in both organizations.

In November 1988 the PDES development work [6], which included the Integrated Product Information Model (IPIM), was presented as a working draft to ISO Committee



STEP/PDES STANDARDIZATION PROCESS

Figure 2

TC184/SC4. The Committee voted to register the working draft with the ISO Secretariat as Draft Proposal (now Committee Draft) 10303 and send it to all participating countries and liaisons for ballot. This international Committee Draft is unofficially named STEP--the Standard for the Exchange of Product Model Data.

It is important to note that the U.S. is not pursuing a separate national standard through the ASME Y14.26 Committee as has been done with versions of IGES. The PDES strategy is to gain international consensus on STEP and then adopt this work under ANSI procedures. This supports the SC4 Committee's development plan of one standard for worldwide use.

Figure 2 depicts the ISO standardization process, the U.S. role in this activity, and the follow-up process of nationalizing the international standard.

ANSI U.S. Technical Advisory Group - The American National Standards Institute (ANSI) is the recognized U.S. representative to ISO and provides the basis for U.S. participation in the international standards activities relating to STEP. To ensure that the positions on standards that are presented to ISO are representative of U.S. interests, ANSI established a mechanism for the development and coordination of such positions. The body which develops national standards in a particular standards area determines the U.S. position in related international standardization activities. Such bodies are designated by ANSI as "U.S. Technical Advisory Groups" (TAGs) for specific ISO activities.

The current U.S. TAG to TC184/SC4 was formed in 1984. Its membership is primarily comprised of technical experts from the IPO. This type of representation ensures that the technical changes that the U.S. believes are necessary are submitted to ISO for consideration at regular TC184/SC4 meetings and through the ballot process. NIST currently serves as Administrator for the TAG.

To foster a better appreciation for the international and national standardization environment in which STEP will play a part, a tutorial on the generic processes is included as an Appendix.

III. Standards Activities Related to STEP or PDES

NIST-Wide Product Data Exchange Task Group - NIST founded the NIST-Wide Task Group to facilitate its work in providing integrated and effective service to U.S. technical and industrial communities who are recognizing the importance of the emerging technology and standards associated with electronic product data exchange to U.S. competitiveness and defense preparedness. The membership of the Task Group consists of technical-professionals actively engaged in research, technical development, standards adoption, and validation testing of product data exchange specifications, implementations, and applications. Members represent the spectrum of NIST interests.

Digital Representation of Product Data Standards Harmonization Organization - This is a new organization reporting to ANSI [7]. The goal of the organization is to facilitate the development of electrical and electronics digital product data standards, with initial emphasis on electrical/electronic product data and its relationship to STEP. The Executive Board is composed of representatives from organizations that develop and approve electrical and electronic product definition standards. In addition to the Executive Board, the organization consists of a Business Needs and Planning Council, a Standards Development Coordination Council, and a Tools and Technology Council.

NIST's Center for Electronics and Electrical Engineering plans to establish a testbed to support the testing requirements identified by the Tools and Technology Council. In addition, NIST is responding to needs for a government-sponsored electronics components data library. Workshops are being scheduled to explore topics such as applicable standards, commercial interests, and existing Department of Defense (DoD) projects.

Navy/Industry Digital Data Exchange Standards Committee (NIDDESC) - NIDDESC is a cost-sharing venture between private firms and government organizations [8]. This effort arose from the Naval Sea Systems Command (NAVSEA) in cooperation with the National Shipbuilding Research Program. The members include leading professionals in the marine industry from major design firms, private shipyards, naval shipyards, and government laboratories. All members are directly involved in CAD/CAM (computer aided design/computer aided manufacturing) in their organizations and together represent a broad spectrum of experience and perspectives.

NIDDESC has subcommittees devoted to specific areas of digital data transfer. The basic objectives are to develop an industry-wide consensus on product data models for ship structure and distribution systems, and on the digital exchange of product model data. Efforts include contributions to IGES, STEP, and the PDES activity, preparation of a Recommended Practices Manual, and the analysis of ship production data flows. NIDDESC has made contributions to the development of DoD standards including MIL-STD-1840, MIL-M-28001 (SGML: Standard Generalized Mark-up Language), and MIL-D-28000 (IGES).

ASME Standards Development Committee Y14 - This is an ANSI-accredited standards making body of the American Society of Mechanical Engineers (ASME). It develops standards for engineering drawings and related documentation practices. Some versions of IGES are submitted for national standards approval via Subcommittee 26, Computer-Aided Preparation of Product Definition Data.

Mathematical Definitions of Dimensions and Tolerances - The Y14 Committee formally established working group Y14.5.1 at its meeting in May 1990. This group has been meeting as an *ad hoc* committee for over a year. It is working on a document that "presents a mathematical definition of geometrical dimensioning and tolerancing consistent with the principles and practices of ANSI Y14.5, Dimensioning and Tolerancing, enabling determination of actual values." A draft standard is expected to be issued within a year.

Apparel Product Data Exchange Specification - The Computer Integrated Manufacturing Committee of the American Apparel Manufacturers' Association has been approved by ANSI as an accredited standards-setting organization. The Committee has established a working group on neutral file representation of apparel product data. Plans call for establishing a short-term guideline of using as a *de facto* standard one of the commonly used proprietary formats, but to develop extensions to STEP for apparel as the long-term solution. The work at this point is focused on establishing functional requirements for an exchange format: basically, how much of the apparel life-cycle should be supported. NIST, under sponsorship of the Defense Logistics Agency, is supporting this effort. NIST has utilized the STEP methodology in preparing a draft specification for 2D pattern making [9].

Application Interface Specification (AIS) - The CAM-I (see page 12 for CAM-I description) Product Modeling Project is trying to establish the AIS as a standard interface to product modelers (formerly known as geometry modelers). AIS was recently redefined in the STEP specification language (Express), and there are strong efforts within both CAM-I and the STEP community to integrate AIS with the STEP Data Access Interface Specification (SDAIS). AIS will be released under ANSI procedures as a draft standard for trial use.

Dimensional Measurement Interface Specification (DMIS) - DMIS, developed by the CAM-I Quality Assurance Project, was approved recently as an ANSI standard. It specifies an APT-like language for two-way communication of inspection information between inspection equipment and a CAD environment. DMIS includes a substantial amount of geometry and tolerance definition, which overlap STEP capabilities, as well as control and measurement result constructs. CAM-I now plans to reorient DMIS around its Application Interface Specification, which will bring DMIS much closer to STEP.

IV. Testing Activities

There are two types of testing activities: 1) testing a standard itself and 2) testing implementations to a standard [10]. The National PDES Testbed, PDES, Inc., Computer-aided Acquisition and Logistic Support (CALS) Test Network, and the IPO Testing Project's Application Validation Methodology Committee are all involved with testing product data standards.

Conformance testing is the testing of a candidate product for the existence of specific characteristics required by a standard specification. It includes testing the extent to which an implementation is a conforming implementation [11]. ISO TC184/SC4/WG6 on Conformance Testing is working to establish conformance criteria for STEP [12]. Three subcommittees of the IPO Testing Project (Testing Methodologies, Interoperability Testing Methodology, Test Case Development), NIST Federal Information Processing Standards (FIPS) testing and the National PDES Testbed, are providing means of testing implementations to the standard through conformance testing.

National PDES Testbed - NIST established the National PDES Testbed at its Gaithersburg, MD site in 1988 to support PDES/STEP development activities. Under the sponsorship of the DoD CALS program, the Testbed has assumed a critical role in the development of STEP. The goal of the Testbed is to provide technical leadership and a testing-based foundation for the rapid and complete development of a STEP standard [13].

The staff of the National PDES Testbed recognize that establishing a STEP standard is very much a consensus-building process. It can only be achieved with support from, and cooperation among standards organizations, industry, government, and academia. The National PDES Testbed is working closely with representatives from all of these different sectors.

Some of the major objectives of the National PDES Testbed are to

- Identify the types of computer applications which will use STEP and model the data used by the applications,
- Specify the technical requirements of these systems with respect to STEP,
- Validate that the STEP specifications satisfy the technical requirements of those systems,
- Design and implement prototype systems which support testing and provide a foundation for future development efforts,
- Maintain control over the many versions of specifications, software tools, and test procedures/data generated by the standards and technical development activities,
- Improve communication and interaction between the various programs and organizations which have a stake in the development of STEP,
- Develop conformance testing procedures which can be used by independent testing laboratories.

PDES, Inc. - In April 1988, several major U.S. technology companies incorporated as PDES, Inc. [14] with the specific goal of accelerating the development and implementation of STEP. In August 1988, the South Carolina Research Authority (SCRA) was awarded the host contract to provide technical management in the implementation of the program. The technical participants provided by the PDES, Inc. member companies and SCRA's subcontractors are under the direction of the PDES, Inc. General Manager from

SCRA. NIST is a government member and provides a testbed facility and technical team members to support the PDES, Inc. effort. PDES, Inc. maintains continual coordination with the IPO.

PDES, Inc. has a multi-phased plan for the acceleration of STEP development. Phases I and II of the plan are each defined to be eighteen-month efforts. The Phase I focus was on testing and evaluating a subset of the December 1988 Committee Draft. The emphasis of the testing and evaluation effort has been placed on a data exchange implementation of mechanical parts and rigid assemblies. Phase II, which began in March 1990, focuses on identifying software implementation requirements, construction of prototypes, development of additional context-driven integrated models (CDIMs) for small mechanical parts, and broadening the program scope to include such areas as electronics, sheet metal, and structures.

CALS Test Network - In January 1988, the Air Force was tasked to take the lead in planning and coordinating testing of CALS standards. The DoD CALS Test Network (CTN) was established, with lead management at the Air Force Logistics Command [15]. The goals of the CTN are to test, evaluate, and demonstrate thoroughly the use of CALS standards in ways that will support accelerating their use and implementation into the digital technical data delivery process for DoD's weapon systems. The CTN performs user-application testing and supports creation of other testing capabilities. The CTN has evolved into a 90-member confederation of industry and government agencies dedicated to supporting that goal.

IGES Testing - IGES has had no conformance requirements in the specification, however the IPO Testing Project has introduced conformance requirement language in IGES, Version 5.0.

NIST sponsored the Society for Automotive Engineers to conduct a testing program for IGES. The IPO Testing Project is evaluating the program's merits and initial testing results, while the IPO Steering Committee is considering "next steps" for IGES testing.

Under the auspices of the CALS Test Network, MIL-D-28000 [16] Class I and II drawings are to be tested by the David Taylor Research Center in 1990, while developing a master test plan for MIL-D-28000 for future years.

Federal Information Processing Standards (FIPS) - The National Computer Systems Laboratory (NCSL) at NIST participates in the development of U.S. Government-wide standards for computer software, hardware, data management, networks, and security. NCSL works through voluntary industry standards organizations to develop standards that will meet the needs of Government users and can be implemented in off-the-shelf commercial products. Those standards that promise sizable benefits to the Government are issued as FIPS.

FIPS and the specifications they adopt are implemented into computer products. Through past experience in research and testing, NCSL sees a need for expansion of its efforts in structuring conformance testing to these FIPS, and is in the process of formulating a program.

SGML (Standard Generalized Mark-up Language) and SQL (Structured Query Language) are two standards being utilized by PDES activities. They can be used as examples of how NIST can support the testing of standards:

SGML - As a 1987 CALS deliverable, NIST developed an SGML validation suite and reference parser. These are both public domain and available through the National Technical Information Service of the Department of Commerce [17]. A Committee in ANSI X3V1 is currently developing standardized test cases under the "conformance testing initiative." These test cases, as they are approved, will also be publicly available as a reference application to test for conformance to SGML.

SQL - A test suite, version 2.0, was released in January 1990. This suite tests compliance with FIPS 127, Database Language SQL [18]. Approximately 60 vendors, integrators, standards organizations, and certification agencies presently use the SQL test suite, Version 2.0, or its predecessor, Version 1.2 (which was released May, 1989) [19]. A commercial conformance testing service began at NIST in April 1990.

VI. International Programs

Commission of the European Communities (EC) - The EC is acting swiftly and deliberately to turn the twelve European countries into a single, integrated market of 320 million people by the end of 1992. The basis for this effort is a 1985 EC White Paper entitled "Completing the Internal Market." That paper sets a timetable for the measures needed to ensure the free circulation of persons, products, service, and capital among the twelve member states. The EC already initiated a program to eliminate the many differing national standards and technical regulations; it has drawn up more than 200 EC directives aimed at harmonizing the various national requirements [20].

The task of establishing European technical standards for products will be left to European standardization bodies set up by industry and other European governments. In the absence of standards to be harmonized or cited, these organizations are to develop standards based on international standards developed by such groups as the International Organization for Standardization (ISO) and the International Electrotechnical Commission (IEC).

Conformance Testing Services (Phase) Two - In support of the EC economic goal of 1992, the United Kingdom, Germany, and France are collaborating on a common objective for CAD/CAM systems data exchange interfaces: to provide harmonized conformance testing services within Europe for the transfer of product models between different CAD systems using neutral formats. Started in December 1988, this service will be ready for use in time for the approval of ISO STEP and will offer interim services for IGES (US-based), VDAFS (German-based) and SET (French-based). Both the development of the test tools and the establishment of operational Testing Laboratories offering testing services are covered in the project. In addition to the three full partners, there is one associate partner, Sweden, a non-EC participant. Beyond national commitment of resources and money, the EC members are funded by the Commission of the European Community.

Established as a multi-phased project, the current phase is commonly known as CTS2, Conformance Testing Services (phase) Two [21]. This is a three-year program completing its second year. {NOTE: The IPO Testing Methodologies Committee under the IPO Testing Project committed to harmonize the procedures and policies being developed under the CTS2 banner. This commitment was made at the April 1990 meeting.}

European Strategic Programme for Research and Development in Information Technology (ESPRIT) - The EC defined ESPRIT after a thorough analysis undertaken in close liaison with industry in 1982 and 1983 [22]. ESPRIT has the following three objectives:

- Provide European Information Technology industry with the basic technologies to meet the competitive requirements of the 1990s;
- Promote European industrial cooperation in Information Technology; and
- Pave the way for standards.

The first phase of the ESPRIT Program started in 1984. The total R&D efforts of the first phase amounted to 1,500M ECU (1 ECU = $1.10), 50% of which were born from the Community budget, the other 50% by the participants in the Program.

The Program is implemented by projects selected from public calls for proposals and based upon the annually updated Workplan. The Program comprises collaborative pre-competitive research and development projects, carried out across frontiers by community companies, universities, and research institutes.

The second phase was initiated in 1988. It is a larger scale operation, but still uses the same mechanism as before: the cost-sharing between the Community and partners, the consensus-building, the Workplan and the on-going assessment principle. The total for this phase is 3,2000M ECU, of which 50% is allocated from the research budget of the Community - about 5% of the R&D expenditure in the Information Technology industry. However, relative to the precompetitive part of R&D, its share is much larger, and it has played an important catalytic role in stimulating a growth of total R&D investment.

STEP/ENV (Draft European Standard) - This is a CEN/CENELEC (European Committee for Standardization/European Committee for Electrotechnical Standardization) project to

- Establish European requirements for implementation and use of STEP,
- Define a framework for European implementation of STEP and any necessary companion standards, and
- Publish an ENV based on STEP.

Participants are Belgium, Denmark, Finland, France, Germany, Netherlands, Norway, Sweden, and the UK [23] .

STEP/ENV is not intended to be different from STEP in any way -- it is a mechanism to allow the STEP standard to be published and harmonized throughout Europe. The project's current mandate finishes in September 1990. STEP/ENV was to be prepared and published in July 1990. It was to consist of a 'cover page' attached to whatever is available from ISO TC184/SC4 in Committee Draft or Technical Report form after the June, 1990 Goteborg meeting.

Advanced Project for European Information Exchange (APEX) - This is a five-year program which began in 1986 with an expected total expenditure of $30M (U.S.). Its members are primarily aerospace industries from Italy, France, the UK, Spain, and Germany. There are four projects: procurement, email, product support of technical documentation, and design engineering. The objective is to develop, qualify and match methodology to products and in turn to services, all based on ISO standards. The design engineering project manager is now (since Paris, 1/90) attending ISO/STEP meetings to coordinate with ISO TC184/SC4 activities.

Japanese STEP Translator Development Project - This is a four-year project that was launched in FY90. The total anticipated cost is $6M (U.S.). It has not been determined (as of the date of this paper) whether they will be writing prototypes or products.

Computer-Aided Manufacturing - International (CAM-I) - CAM-I is an international not-for-profit consortium whose purpose is to advance computer-based manufacturing technology and standards [24]. Its work primarily consists of funding research projects and standards activities. Participation in CAM-I activities generally requires that the organization join the consortium and pay a general membership fee. Member organizations send representatives to periodic CAM-I meetings to establish the technical direction of CAM-I programs.

Current STEP-related programs in CAM-I include Advanced Numerical Control, Quality Assurance, Product Modeling, and Process Planning. CAM-I has shepherded the DMIS (see page 7 for more description) into its adoption as an ANSI standard. CAM-I is actively pursuing adoption of its AIS (see page 7) as a standard software interface to CAD/Solid Modeling systems. To this end, representatives from CAM-I have been participating in STEP committee meetings in a harmonization effort.

A European Computer Integrated Manufacturing Architecture (AMICE) - The objective of project 688 of the ESPRIT Consortium AMICE is to design an open systems architecture for CIM: CIM-OSA (Computer Integrated Manufacturing - Open System Architecture) [25][26]. Twenty-one major European companies in AMICE have joined to produce this framework for enterprise integration in manufacturing. The proposal they have submitted to the ISO committee on Systems Integration and Communication (TC184/SC5/WG1 [27]) consists of three dimensions in a 3X3X4 arrangement. One dimension goes from generic to particular. The second dimension represents three stages in the development process from design to implementation. The final dimension represents four different views from the functional to the organizational.

VII. Summary

This report has briefly described the roles, relationships and technical goals of the current standards and testing activities that relate to product data exchange using STEP. A bibliography is given as a resource if more detailed information is desired about one or more of these activities.

Acknowledgements

The authors would like to acknowledge the following individuals for their contributions to this document: Dave Bettwy, Howard Bloom, Bill Conroy, Jesse Crusey, Jim Fowler, Ted Hopp, Richard Jackson, Tom Leedy, Chuck McLean, Curt Parks, Steve Ray, Kent Reed, Gaylen Rinaudot, Mark Skall, Brad Smith, Frank Spielman, Selden Stewart, and Joan Sullivan.

References

[1] "Overview and Fundamental Principles," Working Draft 10303-1, ISO TC184/SC4/WG6 N6, August 1990.

[2] Smith, B., "External Representation of Product Definition Data," NISTIR 89-4166, September 1989.

[3] "IGES/PDES Organization Reference Manual," National Computer Graphics Association, October 1990.

[4] "Initial Graphics Exchange Specification (IGES) 5.0," NISTIR 4412, National Computer Graphics Association, September 1990.

[5] Smith, B., "Product Data Exchange: The PDES Project - Status and Objectives," NISTIR 89-4165, September 1989.

[6] Smith, B. and Rinaudot, G., "Product Data Exchange Specification: First Working Draft," NISTIR 88-4004, National Technical Information Service (NTIS), Order PB 89-144794, 1988.

[7] "By-Laws of Digital Representation of Product Data Standards Harmonization Organization," Industrial Automation Planning Panel (IAPP), American National Standards Institute (ANSI), 1990.

[8] Billingsley, D., Kloezoi, J., "The NIDDESC: Meeting the Data Exchange Challenge Through a Cooperative Research Program," Naval Shipping Research Program, 1989.

[9] Lee, Y., "On Extending the Standard for the Exchange of Product Data to represent Two-Dimensional Apparel Pattern Pieces," NISTIR 4358, June 1990.

[10] Mitchell, M., Yang, Y., Ryan, S., Martin, B., "Data Model Development and Validation for Product Data Exchange," NISTIR 90-4241, May 1990.

[11] Kemmerer, S., NBSIR 88-3768, "Standards Conformance Testing: Issues and Activities," April 1988.

[12] International Standard Organization, Working Draft 10303-31, "Conformance Testing Methodology and Framework; General Concepts," ISO TC184/SC4/WG6 Document Number N5, July 1990.

[13] McLean, C., "National PDES Testbed Strategic Plan 1990," National PDES Testbed Report Series, NISTIR 4438, October 1990.

[14] For more information contact: PDES, Inc., South Carolina Research Authority, Trident Research Center, 5300 International Blvd., N. Charleston, SC 29418.

[15] "CALS Test Network Strategic Plan," CTN Report 89-008, Lawrence Livermore National Laboratory, Livermore, CA, October 5, 1989.

[16] "Data Representation for Communication of Product Data: IGES Application Subsets," MIL-D-28000, 1990.

[17] "The SGML Parser," NTIS Publication # PB 87-235115/WCC.

[18] "Database Language SQL," FIPS Publication 127-1, February 2, 1990. Available from National Technical Information Service (NTIS).

[19] Sullivan, J., "NIST SQL Test Suite Version 2.0," December 1989. Available from Joan Sullivan, NIST.

[20] Cooke, P., "A Summary of the New European Community Approach to Standards Development," NBSIR 88-3793-1, August 1988.

[21] "Technical Report Number 1: Interim Report on Global Test Concepts (Tasks A-E)," Document CTS2/89/A/005/c, Revision 2[1], October 3, 1989.

[22] European Strategic Programme for Research and Development in Information Technology, "The Project Synopses, Computer Integrated Manufacturing," Vol. 6 of a series of 8, September 1989.

[23] Presentation by Julian Fowler of the CAD/CAM Data Exchange Centre, Leeds University, United Kingdom, May 16, 1990.

[24] For more information contact: CAM-I, 1250 E. Copeland Rd., Ste. 500, Arlington, Texas 76011-8098, U.S.A.

[25] Panse, R., "CIM-OSA - A Vendor Independent CIM Architecture," Proceedings of CIMCON '90, NIST Special Publication 785, U.S. Govt. Printing Office, p. 177, May 1990.

[26] Beckman, D., "CIM-OSA - An Illustrative Example of How to Apply the Modeling Framework," Proceedings of CIMCON '90, NIST Special Publication 785, U.S. Govt. Printing Office, Washington, DC, p. 197, May 1990.

[27] "Reference Model for Shop Floor Production Standards," Draft Paper, ISO TC184/SC5/WG1. Available from CAM-I, Arlington, Texas.

Additional Reading

Bloom, Howard M., "The Role of the National Institute of Standards and Technology as it Relates to Product Data Driven Engineering," NISTIR 89-4078, April 1989.

Frenkel, K., "The Politics of Standards and the EC," Communications of the ACM, Vol. 33, No. 07, p. 41, July 1990.

Gibbons, A., O'Connell, L. "Information About Standards Development for Support of Information Management, Data Sharing, and Data Exchange," DRAFT, Revised June 1990. Available from L. O'Connell, Sandia National Laboratory, Div. 2811, P.O. Box 5800, Albuquerque, NM 87175.

Henghold, W., Schumaker, J., and Baker, L., "Considerations for the Development and Implementation of PDES within a Government Environment," Manufacturing Technology Directorate, Wright Aeronautical Laboratory, Air Force Systems Command, Wright-Patterson AFB, OH, AFWAL-TR-89-8009, February 1989.

Wood, J., Winner, R., "The Relationship Between CALS and Concurrent Engineering," Institute for Defense Analysis, IDA PAPER P-2306, 1990.

Appendix

National and International Standards Processes

ISO Procedures. (source: Cargill, Carl F., Information Technology Standardization Theory, Process, and Organizations, Digital Equipment Corporation, Digital Press, USA, 1989.)

The International Organization for Standardization (ISO) was established in 1946, as a completely voluntary organization. The purpose of ISO is to facilitate the international interchange of goods and services, and to encourage cooperation in economic, intellectual, technological, and scientific endeavors. It is this broad range of interests and concerns that qualifies ISO as the premier standards organization in the world.

The process of creating an international standard is relatively lengthy, but it ensures that consensus is reached. However, in the case of a contentious or complex subject, especially, the leadership and the membership of the originating committee become vitally important if the proposed standard is not to die in internecine wars.

A proposal for a new work item (NWI) is drafted by a P-member, SC, or liaison organization and submitted to a TC for ballot for acceptance. If it passes this first hurdle, it is assigned to an SC for standardization activity. Usually, a WG of the SC creates a Working Draft (WD) and forwards it to the SC for vote. If the SC vote indicates that consensus has been achieved, the WD becomes a Committee Draft (CD), which is then registered and circulated to the full TC. If there is consensus on the CD, it is forwarded to the Central Secretariat for registration as a Draft International Standard (DIS). The Central Secretariat verifies that the DIS meets the requirements of an ISO standard and then circulates it among ISO members for their review and approval; again, consensus is of paramount importance. After receiving the approval of a majority of TC members and 75 percent of ISO voting members, the standard is submitted to the Council for publication as an International Standard (IS).

ANSI Procedures. (source: Cargill, Carl F., Information Technology Standardization Theory, Process, and Organizations, Digital Equipment Corporation, Digital Press, USA, 1989.)

ANSI is the primary interface with the U.S. government on matters relating to standards, providing a mechanism by which the organization and company members can make their needs and desires known to the legislative and regulatory agencies. ANSI is also the recognized representative to ISO and the IEC (International Electrotechnical Commission), the two major international nontreaty standards organizations. The following process outlined, is generalized and may vary depending on the standards group involved. It does give an idea of the checks and balances that are built into the system. In this description, the term "Accredited Standards

Committee" (ASC) will be used to refer to Accredited Organizations and Accredited Canvass groups as well as ASC's. Thus, "ASC" refers to the three types of accredited standards-developing organizations.

A standard begins as a Project Proposal, which may be developed and submitted to an ASC by any individual. The proposal usually is reviewed within the ASC to determine if it fits within the mission of the ASC, as well as to review economic and functional validity. If the ASC believes that the proposal is valid, the full committee, or a representative portion, votes on it. If the vote is positive, the proposal is assigned to an appropriate ASC subcommittee, and the ASC Secretariat usually issues a press release soliciting technical contributions and membership. The subcommittee then develops a calendar and work plan and begins work on the proposal. When the draft proposed American National Standard (dpANS) is completed, the ASC community votes to determine if the document is ready for public review. The dpANS is also liable to review by an ASC subcommittee to verify that it complies with the approved proposal that started the development process. After the ASC review, the dpANS is ready to be forwarded for open public review.

The ASC Secretariat initiates public review of the dpANS. The ANSI publication Standards Action provides a brief description of the standard and announces that it is available for review and comment for the next four months. (It is during this phase that the concept of consensus comes into critical play.) Once consensus is reached, the full ASC votes on the proposal via letter ballot. The ASC Secretariat then forwards the proposal to the ANSI Board of Standards Review (BSR), which verifies that due process was observed and that consensus was, in fact, achieved. When the BSR is satisfied, the dpANS is approved and published as an American National Standard.

The National to International Link. (source: Procedures for U.S. Participation in the International Standards Activities of the ISO, Approved by ANSI Board of Directors, March 19, 1986.

Participation in international standards activities of interest to members of ANSI requires membership in the two nontreaty standardization organizations, ISO, and IEC. For discussion on PDES, NIST will address the relationship specifically between ANSI and ISO, of which ANSI is the U.S. member body. To assure that positions presented to ISO are representative of U.S. interests, a mechanism must exist for the development and coordination of such positions. ANSI normally looks to the body which develops national standards in a particular standards area to determine the U.S. position in a similar international standardization activity. Such bodies are designated by ANSI as "USA Technical Advisory Groups" for specific ISO activities. Where no national standards group exists, or is available to serve, or where several separate national standards groups exist, special bodies will be established for this purpose. (See page 4 for information specific to PDES.)

(부록4)

Companion standard for NC

DRAFT INTERNATIONAL STANDARD ISO/DIS 9506-4

ISO/TC/SC 184/SC 1 Secretariat DIN

Voting begins on
1991-xx-xx

Voting terminates on
1992-xx-xx

INTERNATIONAL ORGANIZATION FOR STANDARDIZATION • МЕЖДУНАРОДНАЯ ОРГАНИЗАЦИЯ ПО СТАНДАРТИЗАЦИИ • ORGANISATION INTERNATIONALE DE NORMALISATION

English title Industrial Automation Systems -
Manufacturing Message Specification (MMS)
Part 4: Companion Standard for Numerical Control.

Titre français Systèmes d'automation industrielle -
Spécification de messagerie industrielle
Part 4: Norme de code a compagnon pour la
machine command numerique

THIS DOCUMENT IS A DRAFT CIRCULATED FOR COMMENT AND APPROVAL. IT IS THEREFORE SUBJECT TO CHANGE AND MAY NOT BE REFERRED TO AS AN INTERNATIONAL STANDARD UNTIL PUBLISHED AS SUCH.

IN ADDITION TO THEIR EVALUATION AS BEING ACCEPTABLE FOR INDUSTRIAL, TECHNOLOGICAL, COMMERCIAL AND USER PURPOSES, DRAFT INTERNATIONAL STANDARDS MAY ON OCCASION HAVE TO BE CONSIDERED IN THE LIGHT OF THEIR POTENTIAL TO BECOME STANDARDS TO WHICH REFERENCE MAY BE MADE IN NATIONAL REGULATIONS.

Contents

Page

1 Scope and Field of Application 1
2 Normative Reference 1
3 Definition 2
3.1 Alarm 2
3.2 Device 2
3.3 Numerical Control (NC) 2
3.4 Numerical Controller 2
3.5 NC-CS (NC Companion Standard) 2
3.6 NC-CS user 2
3.7 NC system 2
3.8 Operator Interface Device 2
3.9 Geometric Definitions 2
4 Abbreviation 3
5 NC Application Description 3
5.1 NC Specific Model 3
5.1.1 NC System State Model 5
5.1.2 The NC Device Control Model 6
5.1.3 NC Information 11
5.1.4 Offset model 12
5.1.5 NC Alarm Model 15
5.1.6 Switching Model 17
5.1.7 NC Data Store Model 17
5.2 NC Specific Functions 18
5.2.1 Identify Functions 18
5.2.2 Data Transfer 18
5.2.3 NC Data Store Management 19
5.2.4 Process management 19
5.2.5 Interaction with an operator 19
5.2.6 Alarm Processing 19
5.2.7 Switching Operation 19
6 NC Application Specific Context Mapping 19
6.1 Mapping of the NC specific Model to the VMD Object 19
6.1.1 Mapping of the NC system to the VMD Model 20
6.1.2 NC VMD 22
6.2 Definition of NC specific Objects which Map to Domains 22
6.2.1 NC Domain Object 22
6.2.2 Device Domain 22
6.2.3 Process Information Domain 22
6.2.4 Program Domain 22
6.2.5 Data Domain 22
6.2.6 NC Data Store Domain 22
6.2.7 Other Objects mapping to Domains 24
6.3 Definition of NC specific Objects which Map to Program Invocation 24
6.3.1 NC Program Invocation Object 24
6.3.2 Controlling Process Program Invocation 24
6.4 Definition of NC specific Objects which Map to other MMS abstract Objects 28
6.4.1 Definition of NC specific Objects which Map to Named Variables 28
6.4.2 Definition of NC specific Objects which Map to Named Variable List Objects 29
6.4.3 Definitions of NC specific Objects which map to Named Type Objects 29
6.4.4 Definition of NC specific Objects which Map to Event Condition, Event Action and Event Enrollment Objects 30
6.4.5 Mapping of the Switching Model to MMS Objects 30

6.5 Definition of New MMS Abstract Objects to support other NC Specific Objects 31
7 Services 31
7.1 Use of the MMS services 31
7.2 Definition and Use of Application-specific Services 33
7.3 The Initiate Service and Protocol 33
7.3.1 Definition of NC-specific Initiate Service 34
7.3.2 Definition of NC-specific Initiate Protocol 34
7.4 End of Module 34
8 Standardized NC Specific Objects 34
8.1 Domain Objects 35
8.1.1 Device Domain 35
8.1.2 Program Domain 35
8.1.3 Process Information Domain 35
8.1.4 Tool Data Domain 36
8.1.5 Means of Production Data Domain 36
8.1.6 Setup Data 36
8.1.7 Statistical Data 36
8.1.8 Probe Data Table 37
8.1.9 Diagnostic Data 37
8.1.10 Device Origin Setup Domain 37
8.1.11 NC Data Store Domain 38
8.2 Program Invocation Objects 38
8.2.1 Program Invocation Object N_DROP 38
8.3 Named Variable Objects 38
8.3.1 NC VMD-specific Named Variable Objects 39
8.3.2 NC Domain specific Named Variable Objects 39
8.4 Scattered Access Objects 49
8.5 Named Variable List Objects 49
8.5.1 NC VMD-specific Named Variable List Objects 49
8.5.2 NC Domain-specific Named Variable List Objects 49
8.6 Named Type Objects 50
8.6.1 Data Store Entry Type 50
8.7 Semaphore Objects 51
8.7.1 Control Token 51
8.8 Operator Station Objects 51
8.9 Event Condition Objects 51
8.10 Event Action Objects 51
8.11 Event Enrollment Objects 51
8.12 Journal Objects 51
8.13 Other NC Specific Objects 51
9 Conformance 51
9.1 Conformance Class Description 51
9.1.1 Minimum Requirements For The Functional Group Identify 51
9.1.2 Minimum Requirements For The Functional Group Data Transfer 51
9.1.3 Minimum Requirements For The Functional Group NC Information 52
9.1.4 Minimum Requirements For The Functional Group Process Management 52
9.1.5 Minimum Requirements For The Functional Group Operator Interface Device 52
9.1.6 Minimum Requirements For The Functional Group Alarm Processing 52
9.1.7 Minimum Requirements For The Functional Group Switching Operation 52
9.2 Restrictions On MMS Optional Parameters 52
9.3 Conformance To Standardized Objects 52
9.4 Addition to MMS PICS 53
9.4.1 PICS For Functional Classes 53
9.4.2 PICS For Standardized Objects 53

Annexes

A Dynamic Download and Coded Domains . . . 56

A.1 Dynamic Download . . . 56

A.2 Coded Domains . . . 57
A.2.1 Abstract Syntax for Structured Domains . . . 57
A.2.2 Abstract Syntax for Character Based User Programs . . . 58

B Extension of ISO 9506-4 to Include NC Milling Machine Functionality . . . 59

B.0 Introduction . . . 59

B.1 Scope And Field Of Application . . . 59

B.2 References . . . 59

B.3 Definitions . . . 59

B.4 Symbols and Abbreviations . . . 59

B.5 Application Description . . . 59
B.5.1 Tool Magazines, Tool Holders, Tools, Edges . . . 59

B.6 NC Milling Specific Context Mapping . . . 60
B.6.2 Objects Which Map To Domains . . . 60
B.6.4 Objects Which Map To Named Variables . . . 61

B.8 NC Milling Specific Standardized Objects . . . 63
B.8.2 Standardized Domains . . . 63
B.8.3 Named Variable Objects . . . 63

B.9 Conformance . . . 66
B.9.4 PICS For Standardized Objects . . . 66

C Extension of ISO 9506-4 to Include Turning Machine Functionality . . . 67

C.0 Introduction . . . 67

C.1 Scope And Field Of Application . . . 67

C.2 References . . . 67

C.3 Definitions . . . 67

C.4 Symbols and Abbreviations . . . 67

C.5 Application Description . . . 67
C.5.1 Part Setup . . . 68
C.5.2 Taper Trims . . . 68

C.6 NC Turning Specific Context Mapping . . . 68

C.8.3 Named Variable Objects . . . 68

C.9 Conformance . . . 70
C.9.4 PICS for standardized objects . . . 70

D Extension of ISO 9506-4 to Include Flexible System Functionality . . . 71

D.0 Introduction . . . 71

D.1 Scope And Field Of Application . . . 71

D.2 References . . . 71

D.3 Definitions . . . 71

D.4 Symbols and Abbreviations . . . 71

D.5 Application Description . . . 71
D.5.1 Parking Locations, Partitions . . . 72
D.5.2 Pallets, Partitions, Fixtures, Parts . . . 72

D.6 Flexible Systems Specific Context Mapping . . . 74
D.6.2 Objects Which Map To Domains . . . 74
D.6.4 Objects Which Map To Named Variables . . . 74

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work.

Draft International Standards adopted by the technical committees are circulated to the member bodies for approval before their acceptance as International Standards by the ISO Council.

They are approved in accordance with ISO procedures requiring at least 75 % approval by the member bodies voting.

International Standard ISO 9506, Part 4 was prepared by Technical Committee ISO/TC 184 "Industrial Automation and Integration" Sub-Committee SC 1 "Physical Device Control" ISO/IEC 9506, Part 4 is one of several standards as a companion to ISO/IEC 9506 "Manufacturing Message Specification" (MMS), and deals with the communication needs especially between numerically controlled manufacturing systems and host computers. Together with ISO/IEC 9506, Parts 1 and 2 "MMS-Core" and other companion standards,this ISO/IEC 9506, Part 4 enables the networking with other programmable devices in the shop floor.

0 Introduction

Part 1 of this standard, ISO/IEC 9506, defines the MMS services, a wide variety of services which are useful for the interworking of many types of manufacturing and process control devices when used in open communication systems conforming to the OSI Model (ISO 7498). However, these MMS services by themselves only provide a messaging environment for abstract, generic types of such control devices.

In order to convey device-specific semantics for real manufacturing equipment such as NC machines, robots, programmable controllers, process control systems, and so forth, many of the MMS services have optional parameters which have not been detailed in MMS. But MMS also specifies that any of these parameters may only be used in the context of a so-called "companion standard" designed for specific devices by the appropriate, recognized standardizing organization.

The standard herein, Part 4 of ISO/IEC 9506, is intended to be such a companion standard to parts 1 and 2 of ISO/IEC 9506. Specifically, it defines the numerical control semantics to the Manufacturing Message Specification. It should be used when numerically controlled manufacturing systems or devices are connected to a communication network conforming to the OSI Model and employing the MMS service and protocol.

Part 4 of ISO/IEC 9506 hereinafter shall be variously referenced as "this standard", "this international standard", "this part of ISO/IEC 9506", or simply "ISO/IEC 9506-4".

The definitions and specifications of the numerical control specific semantics in this part of ISO/IEC 9506 follow the requirements and guidelines of Annex A of ISO/IEC 9506-1. Thus, this standard has nine clauses as outlined here:

Clause 1 defines the scope of this standard.

Clause 2 lists related OSI standards.

Clause 3 defines all new terms used herein which are in addition to those defined in ISO/IEC 9506-1.

Clause 4 lists the abbreviations used in this standard.

Clause 5 describes the manufacturing environment within which NC applications may operate under this standard. The objective of this clause is to describe the application area in a hierarchy of models which are related to the MMS abstract models described in Clause 6.

Clause 6 defines the NC-specific application context as an extension of the MMS application context.

Clause 7 defines the syntax of all MMS services which have to be defined by this standard.

Clause 8 defines the standardized objects used with this part of ISO/IEC 9506.

Clause 9 contains all of the conformance requirements for implementations adhering to this part of ISO/IEC 9506.

In addition to the above clauses, this part of ISO/IEC 9506 also contains the following annexes, both normative and informative:

Annex A (Normative): Dynamic Downloading and Coded Domains

Annex B (Normative): Extension of ISO 9506-4 to Include NC Milling Machine Functionality

Annex C (Normative): Extension of ISO 9506-4 to Include Turning Machine Functionality

Annex D (Normative): Extension of ISO 9506-4 to Include Flexible System Functionality

Annex E (Informative): Application Examples

NOTE: ISO/IEC 9506-4 does not constrain the mapping from real to virtual numerically controlled manufacturing devices, nor does it specify individual implementations or products. This standard also recognizes that safe operation of NC devices shall be paramount, and that any operations described or defined herein are permissible only if complying with the relevant safety standard(s) and regulations.

Industrial Automation Systems - Manufacturing Message Specification (MMS) Part 4: Companion Standard for Numerical Control

1 Scope and Field of Application

This part of ISO/IEC 9506 extends the concepts and principles defined in ISO 9506, part 1 and part 2.

ISO/IEC 9506-4, as a companion standard to Manufacturing Message Specification, contains the semantics of numerically controlled manufacturing devices and equipment. Specifically, ISO/IEC 9506-4:

a) describes the use of Manufacturing Message Specification services in NC specific applications,

b) describes the model of an NC device in terms of its application specific functions and how these functions map onto the attributes of a Virtual Manufacturing Device (VMD),

c) provides the NC specific syntax for those MMS services which are applicable to NC operations, and which also allow companion standard specific parameters,

d) defines "standardized" names for NC specific objects,

e) defines NC specific conformance in conjunction with the MMS service and parameter conformance building blocks (CBBs) required.

Furthermore, it should be noted that in addition to this part of ISO/IEC 9506, within ISO and IEC other companion standards have been developed or such work is in progress, in the application areas of Robots and Robotic Systems, Programmable Control Systems, and Process Control Systems.

2 Normative References

The following standards contain provisions which apply to this international standard.

ISO 841, *Numerical Control of Machines - Axes and Motion Nomenclature*

ISO/DIS 2806, *Numerical Control of Machines - Vocabulary*

ISO 6983-1, *Numerical Control of Machines - Program Format and Definition of Address words - Part 1: Data format for positioning, line motion and contouring control systems*

ISO 6983-2.2, *Numerical Control of Machines - Program Format and Definition of Address words - Part 2: Coding and Maintenance of preparatory functions G and universal miscellaneous functions M*

ISO 6983-3, *Numerical Control of Machines - Program Format and Definition of Address words - Part 3: Coding of miscellaneous functions M (classes 1 to 9)*

ISO 7498, *Information Processing Systems - Open Systems Interconnection - Basic Reference Model*

ISO TR/8509, *Information Processing Systems - Open Systems Interconnection - Service Conventions*

ISO 8824, *Information Processing Systems - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1)*

ISO 8825, *Information Processing Systems - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1)*

ISO/IEC 9506-1, *Manufacturing Message Specification (MMS) - Service Definition*

ISO/IEC 9506-2, *Manufacturing Message Specification (MMS) - Protocol Specification*

NOTE - Users should recognize that all international standards undergo revisions from time to time and that any reference made herein to any other international standard implies its latest edition, unless stated otherwise.

3 Definitions

The terms defined in the following subclauses are used in various contexts throughout this part of ISO/IEC 9506. Note that some of these definitions also define several of the abbreviations listed in clause 4.

3.1 Alarm

Alarm: The immediate indication or notification of the occurrence of a severe problem (malfunction) or abnormal condition within the NC system, usually causing an interruption of the operation, which should be resolved externally (such as through operator intervention).

3.2 Device

Device: In the context of this standard, "Device" is a term used to describe, as a whole or in part, the machinery or electro-mechanical equipment employed in manufac-turing or other processes. Devices may be machine tools, functional parts of machine tools, workpiece handling equipment, or any similar apparatus.

In this context, a device shall be visible to an NC-CS user as a separate unit which may be controlled independently by the NC-CS user, such as started and stopped.

3.3 Numerical Control (NC)

Numerical Control: Automatic control of a manufacturing or other process performed by one or more device(s), usually involving motion mechanisms, and making use of preprogrammed numeric data, which may be introduced while the operation is in progress.

3.4 Numerical Controller

Numerical Controller: An electronic control system designed to perform the numerical control (NC) functions of manufacturing or other processes. It usually consists of a computer, file store(s), both analogue and digital input and output interfaces, and one or more operator interface device(s).

3.5 NC-CS (NC Companion Standard)

NC-CS: This part of ISO/IEC 9506.

3.6 NC-CS user

NC-CS user: The part of an application process which invokes the NC Companion Standard. In the context of this standard, this refers to the external, or "host", processor which communicates with, and exerts some measure of remote control over the NC system.

3.7 NC system

NC system: A workstation which consists of a numerical controller and one or more devices.

3.8 Operator Interface Device

A operator interface device provides the means for the NC-CS user to communicate with the NC system operator. Conversely, the same interface may be used by the NC system operator to control the NC system in the LOCAL CONTROL mode.

3.9 Geometric Definitions

1) Offset: General term which may refer to such general concepts as zero offset, axes value transformation, coordinate transformation, or geometry transformation.
2) Axis: Generally refers to a real, guiding (linear or rotary) axis of a machine, which also reflects one dimension in the work space of the machine.
3) Coordinate Axes: see Coordinate System.
4) Zero Offset: The relationship (in one dimension) between the origins of a real machine axis and the corresponding part program coordinate.
5) Axis Value Transformation: (One dimensional) transformation (sequence of such primitives as translation, mirroring, and scaling) of a point on a real machine axis in terms of the corresponding part coordinate.
6) Coordinate System: Defined in ISO 841 (Note: always right-handed). The Coordinate System may also be uniquely defined by three orthogonal coordinate axes.

7) Actual Coordinate System: The actual coordinate system used by the NC.

8) Coordinate Transformation: Expression of the spatial relationship (location and orientation) between two independent coordinate systems.

9) Geometry Transformation: The application of a sequence of primitive transformations (translation, rotation, scaling, mirroring) to a point in space in terms of the actual coordinate system.

In addition to the definitions in this clause, the definitions of clause 3 of ISO/IEC 9506, part 1 apply, which in addition to MMS-specific definitions lists a number of terms defined in ISO 7498, the OSI Basic Reference Model, in ISO/TR 8509, the OSI Service Conventions, and in ISO 8824, the Abstract Syntax Notation One Specification. All of these definitions apply to this standard by reference.

Many additional, NC specific definitions may be found in ISO 2806.

4 Abbreviations

The following abbreviations are used in this part of ISO/IEC 9506:

ASN.1	Abstract Syntax Notation One
CBB	Conformance Building Block
Cnf	confirm
CS	Companion Standard, often used with NC: NC-CS
Ind	indication
M	mandatory parameter
MMS	Manufacturing Message Specification
NC	Numerical Control or Numerical Controller
OSI	Open Systems Interconnection
PC	Programmable Controller
Req	request
Rsp	response
VMD	Virtual Manufacturing Device

5 NC Application Description

This chapter describes the model and functionality of an NC system in a manufacturing communication environment.

5.1 NC Specific Model

As defined in clause 3.7, an NC system is a workstation, where one or more devices are controlled by a Numerical Controller (NC). Such devices may consist of any mechanical or electro-mechanical equipment depending on their individual application. An NC system may be connected to other stations through one or more communication channel(s).

In a factory, several NC-CS users may communicate with a single NC system, as shown in figure 1.

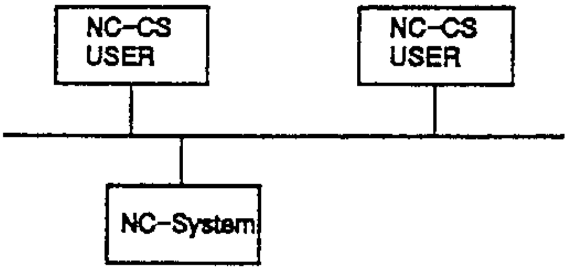


Figure 1 - NC System in the Manufacturing Environment

An NC system may be described by models of its component parts. Figure 2 models the component parts. These component parts may describe aspects of the numerical controller, the device, or both.

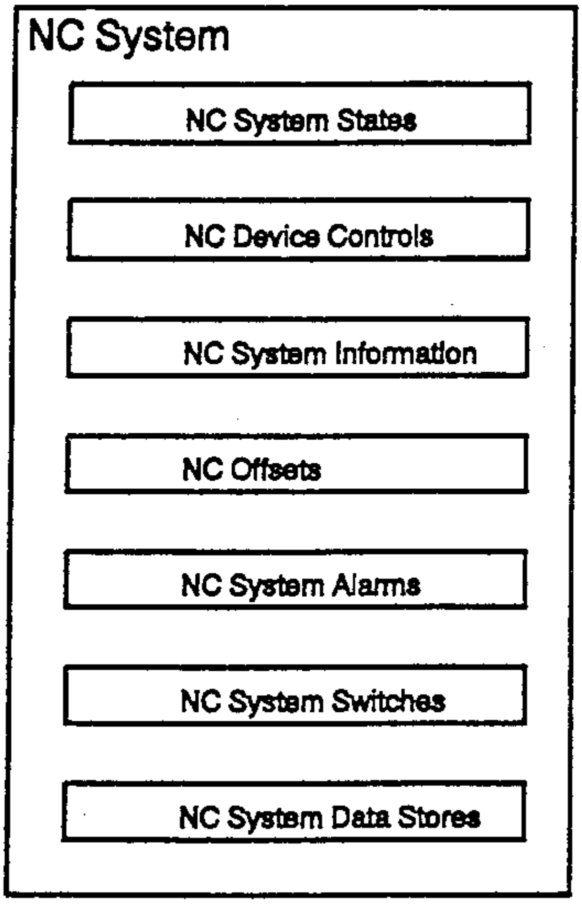


Figure 2 - NC System Components

The models in this section describe the elements of the NC application area from a communication point of view. They include the following models:

- the NC system state model,
- the NC device control model,
- the NC information model,
- the NC offset model,
- the NC alarm model,
- the NC switching model,
- the NC data store model.

5.1.1 NC System State Model

The NC System State Model describes the global state of an NC system, as visible to an NC-CS user. Also, it represents a mechanism which allows coordination between local users and NC-CS users (remote users), as well as the coordination between multiple NC-CS users.

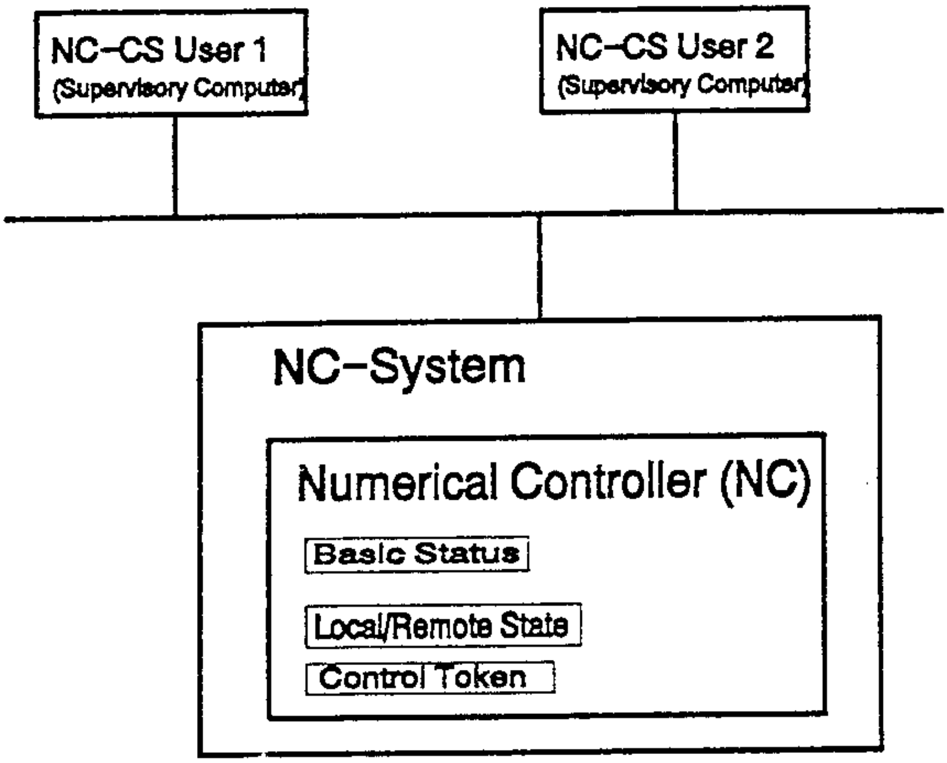


Figure 3 - NC System State Model

5.1.1.1 Basic Status

The Basic Status describes the overall working condition of the NC system.

5.1.1.1.1 State Description

a) NORMAL PRODUCTION - The NC system is fully operable.

b) LOCAL-ACTION REQUIRED - A component of the NC system has failed or is not operable

5.1.1.2 Local/Remote State

In the factory environment an NC system may be controlled by an NC-CS user (remotely) or locally, as shown in figure 4.

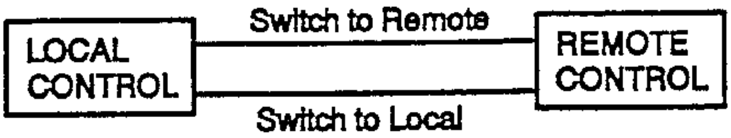


Figure 4 - Local/Remote State

5.1.1.2.1 State Descriptions

a) LOCAL CONTROL - In this state, the NC operation is under control of a local user. For a transition to the REMOTE CONTROL mode, local action by the operator is required. This part of ISO/IEC 9506 does not include any states or operations under LOCAL CONTROL.

NOTE - Even in LOCAL CONTROL some remote operations may be possible. For example, an NC-CS user may inform the NC system about the next part to be made. The choice of remote operations which may be allowed in LOCAL CONTROL is implementation dependent, and should be closely related to security. This choice may even be changed as a result of operator decisions.

b) REMOTE CONTROL - In this state, the NC system is controlled by a remote user. Any operator intervention which modifies the state of the controlling process, or which results in a safety violation shall force the transition to LOCAL CONTROL.

5.1.1.2.2 State Transitions

a) Switch to Remote - This transition shall only be the result of local action and shall enable remote control of the NC system. While this transition may also depend on the state of the Controlling Process, the exact conditions for this shall be at the discretion of the system implementer.

b) Switch to Local - This transition shall change the NC system from REMOTE CONTROL to LOCAL CONTROL. It may take place during any of the Device states and Controlling Process states, and the resultant states of the Device(s) and/or Controlling Process(es) shall be implementation dependent.

5.1.1.3 Control Token

An NC system may be controlled by several NC-CS users. Such a multiple NC-CS user environment requires a mechanism for assuming and relinquishing control.

Only the user in possession of the token shall have full control of the NC system, which shall not respond to requests from other users for actions which affect the Controlling Process states. However, certain status information and machine programs may be available to other users.

NC systems which do not support multiple NC-CS user environments do not require a control token.

5.1.2 The NC Device Control Model

The NC Device Control Model may be viewed as a number of interacting components, such as one or more Devices, a Numerical Controller, an Operator Interface Device, and one or more Controlling Processes, as illustrated in figures 5 through 7.

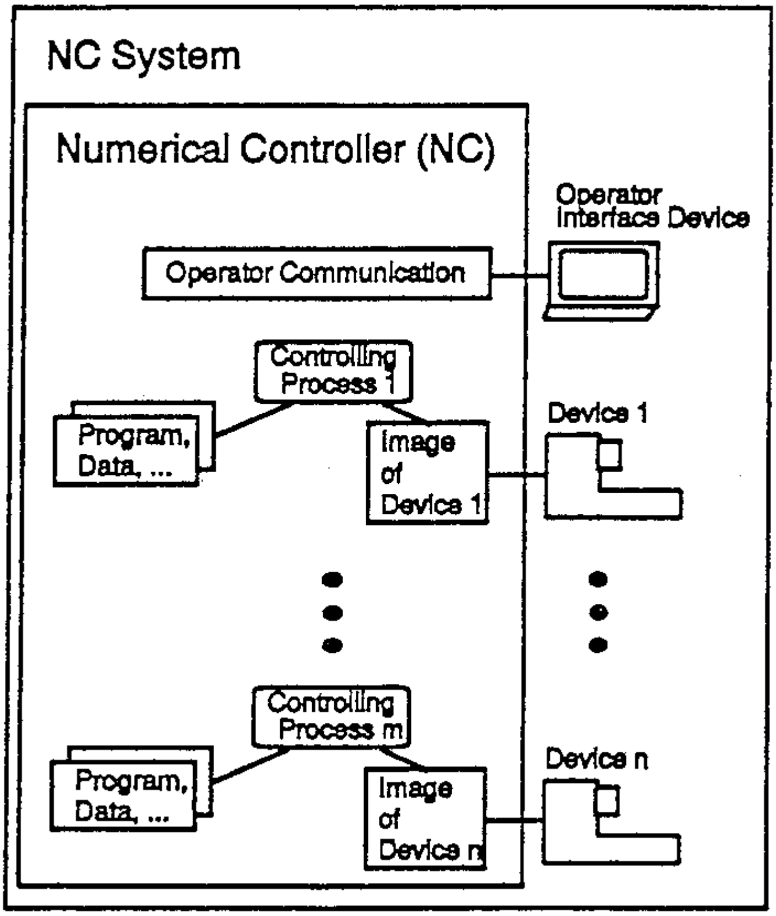


Figure 5 - NC Device Control Model

A) As defined in clause 3.2 of this standard, "devices" represent those mechanical or electro-mechanical parts of an NC system which may be visible to an NC-CS user as a separate units which may be controlled separately, such as started or stopped.

Note - The manner in which devices are delineated within a system shall be arbitrary. While one implementation may consider as devices such overall entities as a milling machine or tool warehouse, another one may subdivide a system into smaller units, such as spindle, rotary table, or tool changer.

B) The numerical controller contains an image of every device within the NC system. This image is used to manage and control the device. It may include infromation about the physical resources (such as number of axes, travel limits, and other configuration data), about the current state (device ready, calibration state, and so on), and about the operating paramters of the device (such as override values).

C) Controlling Processes represent the execution of the user programs, which may include part programs and data such as tool corrections and fixture offsets. They may also include execution information, such as the current execution state, the line of program currently being processed, and so on.
Controlling Processes are linked to devices, and they may be created dynamically by the NC-CS user.

D) A operator interface device provides the means for the NC-CS user to communicate with the NC system operator. Physically, the same interface may also be used by the NC system operator to control the NC system in the LOCAL CONTROL mode.

The general model of an NC system may consist of one or more devices and one or more Controlling Processes. A simple numerical controller, however, may support only a single device and a single Controlling Process.

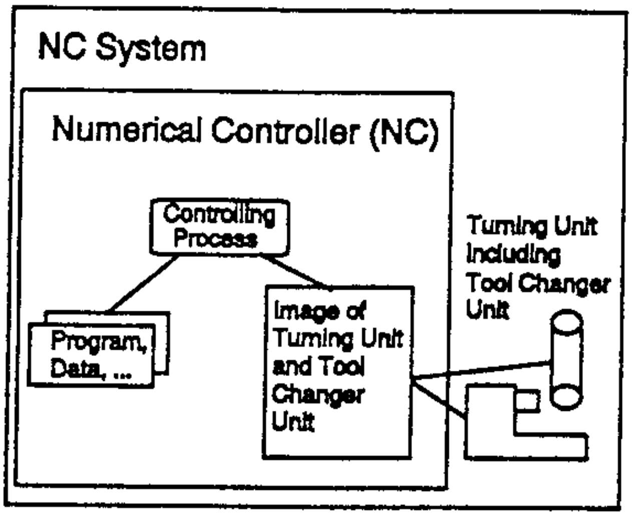

NC System
Numerical Controller (NC)
Controlling Process
Program, Data, ...
Image of Turning Unit and Tool Changer Unit
Turning Unit Including Tool Changer Unit

Figure 6 - NC Device Control Model (Single Device)

Figure 6, for example, shows an NC system consisting of a single device and a single Controlling Process. The device image represents a turning unit and a tool changer. The Controlling Process links programs and data to the device.

Other numerical controllers may be able to control several, usually interacting devices. An example of such a system is shown in figure 7, which models three separate devices, two turning units, and a tool warehouse. The NC-CS user may act on each of these devices independently.

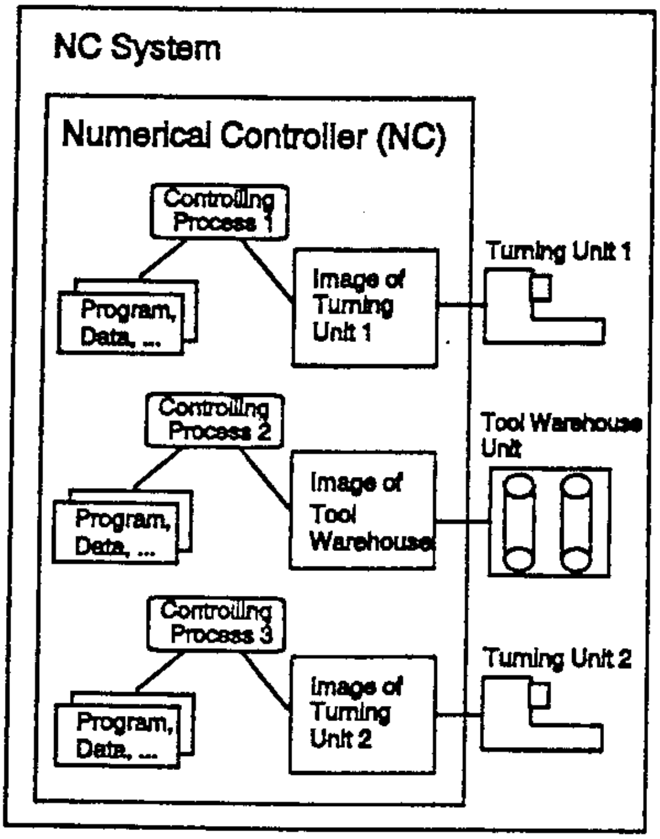


Figure 7 - NC Device Control Model (3 independent devices)

The operation of the NC device is described by the Device State Model and the Controlling Process State Model.

5.1.2.1 The Device State Model

The Device State Model describes the state of the device. The image of the device includes the Device State.

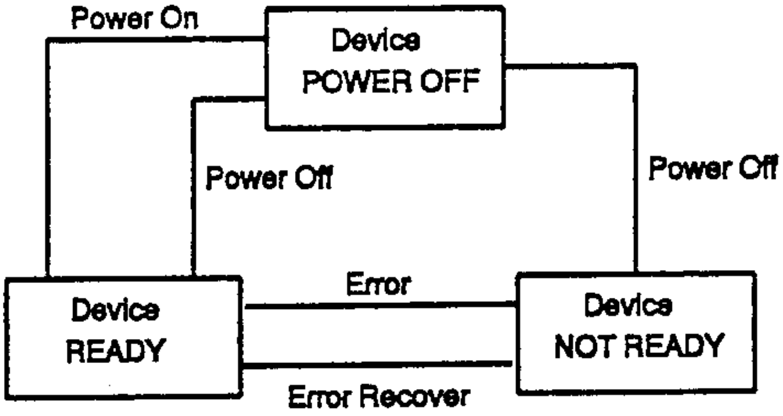


Figure 8 - NC Device State Model

5.1.2.1.1 State Descriptions

a) POWER OFF - In this state, all device functions are inactive except for those pertaining to safety (e.g. brakes, clamps). No power should be applied to the device.

b) READY - In this state, power is applied to the device, that is, axes drives, spindle drives, hydraulic pumps, etc.

c) NOT READY - This state may be caused by an error or a stop condition. If caused by an error condition, only an action which acknowledges and removes the error condition may cause an exit from this state. It is assumed that the NC provides sufficient information for proper diagnosis.

5.1.2.1.2 State Transitions

a) Power On - This is associated with energizing the device, and represents the the transition from POWER OFF to READY.

b) Power Off - This transition puts the device into the POWER OFF state, which shall halt all operations of the device.

c) Error - This transition puts the device from the READY state into the NOT READY state.

d) Error Recover - This transition returns the device into the READY state.

NOTE - The above device state model is simplified. In reality, such transitions as Power On and Power Off may be rather complex procedures.

5.1.2.2 The Controlling Process State Model

A Controlling Process may control one or more devices. For an NC-CS user it is necessary to be aware of the program, and the way the program acts on the devices.
Therefore, the Controlling Process state model describes the execution states of the user program, such as start, stop, and so on.

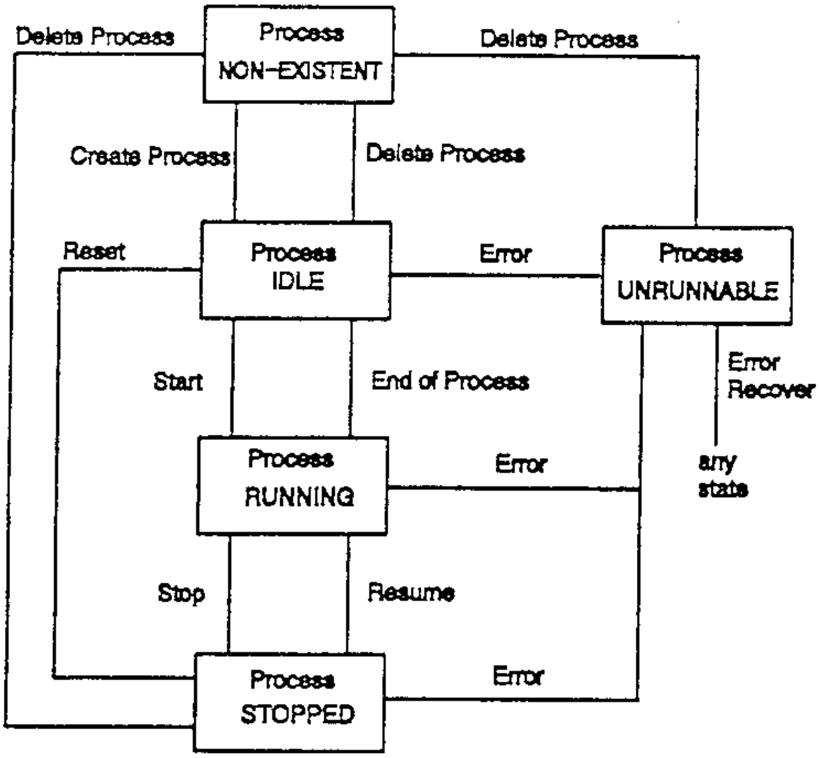


Figure 9 - NC Controlling Process State Model

The Controlling Process may be created dynamically by the NC-CS user. Such creation usually entails the binding of a device to the Controlling Process, and the selection of user programs and data. To operate any device, at least one Controlling Process is needed, and in some cases the binding of such Controlling Process and its device may be static.

5.1.2.2.1 State Descriptions

a) NON-EXISTENT - This is the state of a Controlling Process prior to its creation.

b) UNRUNNABLE - This state shall be the result of an error. Recovery from this state shall require the explicit action of a operator.

c) IDLE - In this state the Controlling Process shall be ready to execute, that is, all required initialization shall have been completed.

d) RUNNING - In this state the Controlling Process shall be fully operational and executing user program commands.

e) STOPPED - This state may be the result of a stop command, or any other stop condition. From this state, after stopping, the user program may resume execution normally.

5.1.2.2.2 State Transitions

State transitions may be initiated from a number of sources, such as the NC-CS user, the operator, the user program, or the Device. Also, it is presumed that the manufacturer may define certain conditions which will inhibit these transitions, while providing sufficient information to properly diagnose the situation.

a) Create Process - This transition creates a Controlling Process by linking user programs, data, and devices.

b) Delete Process - This transition deletes a Controlling Process, by deleting all such links between programs, data, and devices.

c) Start - This transition starts the execution of the Controlling Process, assuming that all prerequisites have been fulfilled.

d) End of Process - This transition completes the execution of the Controlling Process. This transition usually corresponds to the end of the program.

e) Stop - This transition stops execution of the program. It may be caused by a stop command from the NC-CS user, and it puts the Controlling Process into the STOPPED state. Execution may be restarted by the Resume transition.

f) Resume - This transition restores the RUNNING state.

g) Error - This transition shall suspend the Controlling Process and cause it to be UNRUNNABLE. The cause of the error shall be removed before the process may be recovered from this state.

h) Error Recover - Removing the cause of a previous error by local or remote action shall result in this transition from the UNRUNNABLE state to any prior state.

i) Reset - This transition puts the Controlling Process into the IDLE state.

5.1.2.2.3 Correlation To The Device State

Since the states of a device and its associated Controlling Process states are interdependent they will influence each other. For instance, when a device is switched off or enters the NOT READY state, then there should be a Controlling Process state transition (see Table 1).

Table 1 - **Correlation between Device States and Controlling Process States**

Device State	Possible Controlling Process States
POWER OFF	NON-EXISTENT, UNRUNNABLE, IDLE
READY	any state
NOT READY	NON-EXISTENT, UNRUNNABLE, STOPPED, IDLE

5.1.3 NC Information Model

Basic Status, Device Status, and the Controlling Process Status provide the NC-CS user with general information about the state of the NC system. However, to provide the more detailed information about the NC system required for remote control by the NC-CS user, this international standard includes the additional types of information described here by the NC Information Model.

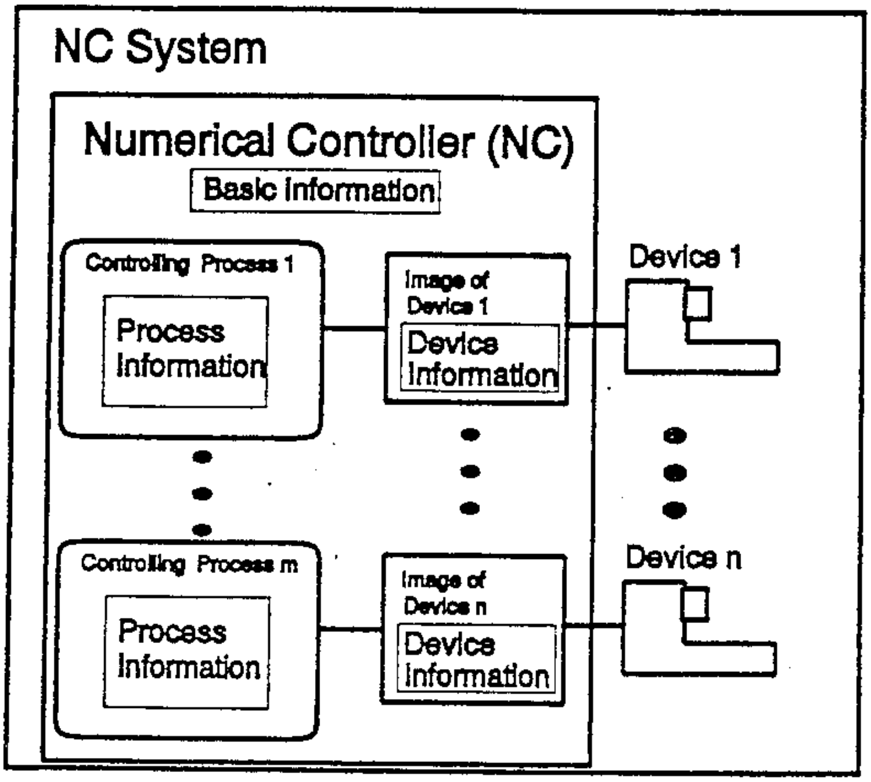


Figure 10 - NC Information Model

5.1.3.1 Basic Information

This encompasses information about the Numerical Controller, such as controller voltage or controller error information.

5.1.3.2 Device Information

This encompasses information about the device, such as hydraulic pressure or device voltage.

5.1.3.3 Process Information

This encompasses information about the Controlling Process, such as the currently executing program block, or current procedure call.

5.1.4 Offset Model

An important part of the operation of an NC system is the capability to handle axis transformation, zero offsets, and other coordinate transformations. In stand-alone, conventional NC systems most of this functionality is handled by the machine operator, as well as by user programs. The purpose of the Offset Model is to provide the NC-CS user with these functions.

The Offset Model shall provide the NC-CS user with all the geometric transformation capabilities required for remote control. Offsets represent the transformations of the location descriptions of points which are usually associated with the machined part or workpiece. The Offset Model distinguishes two types of such transformations, one dimensional and three dimensional, that is, in terms of one axis, or in terms of Cartesian coordinates. Both types further distinguish between (1) transformation of the reference origin, and (2) transformations of axis values or geometry. For better visualization of the Offset Model refer to figures 11 through 15.

For tutorial information on this, see Annex E.

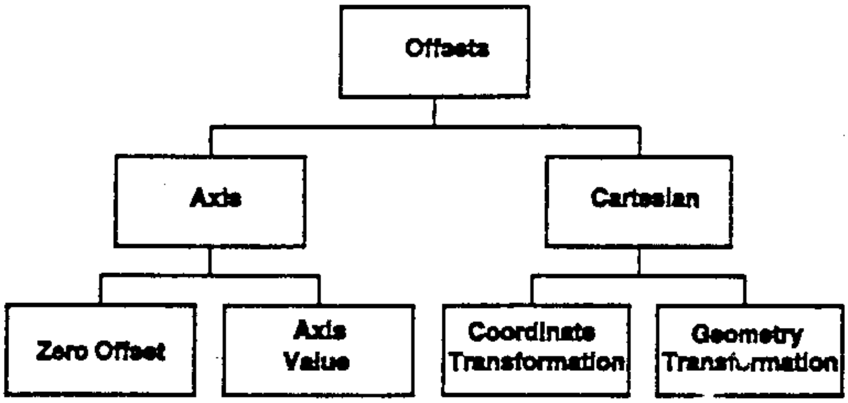


Figure 11 - Hierarchy/Classification of offsets

5.1.4.1 Zero Offset

A zero offset shifts the virtual origin (zero point) of a physical machine axis (see figure 12). This virtual origin is the reference for the part program.

Rotary and linear axes are treated equally.

virtual origin = real origin + zero offset 1 + zero offset 2 ...+ zero offset n

NOTE - The number of zero offsets supported in an NC is implementation specific, and shall be stated by the appropriate PICS entry. For rotary axes, there shall also be the indication of the permissible range (0..360, -180..180, ...).

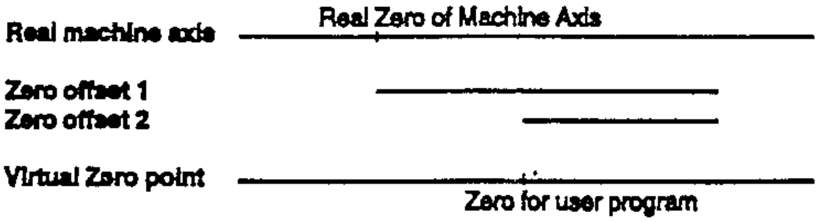


Figure 12 - Zero offset/Virtual zero point

5.1.4.2 Axis Value Transformation

All axis value transformations are with reference to the virtual origin of the respective axis. Rotary and linear axes are treated equally. In the end effect, axis value transformations change the location or geometry of the part or workpiece produced by the user program.

ISO/IEC 9505-4 defines the following primitive axis value transformations (see figure 13):

Translation:

Translation of the point P0 with value x

```
        primitive T(x):  P1 = T(x) * P0   (matrix notation)
                         p1 = x + p0      (coordinate notation)
```

Mirroring:

Mirroring of point P0 about the virtual origin.

```
        primitive M():   P1 = M() * P0    (matrix notation)
                         p1 = - p0        (coordinate notation)
```

Scaling:

Scaling of point P0 relative to the virtual origin with the scale factor x.

```
        primitive S(x):  P1 = S(x) * P0   (matrix notation)
                         p1 = x * p0      (coordinate notation)
```

Scaling and mirroring by themselves, separately, are commutative. Otherwise, any sequence of primitive operations is order dependent. Following is a commonly used notation which reflects the order:

```
P1 = T(2) * M() * S(2) * T(1) * P0
                         ^ first primitive
                  ^ second primitive
           ^ third primitive
   ^ fourth primitive
```

NOTE - This standard does not limit the number of primitive transformations, which shall be implementation specific.

NOTE - The range specification for rotary axes shall be implementation specific (for instance, 0..360, -180..180, ...).

NOTE - Complex axis value transformations may be expressed by a sequence of primitives, such as mirroring about centre point 5, which may be achieved by the following expression:

P1 = T(5) * M() * T(-5) * P0

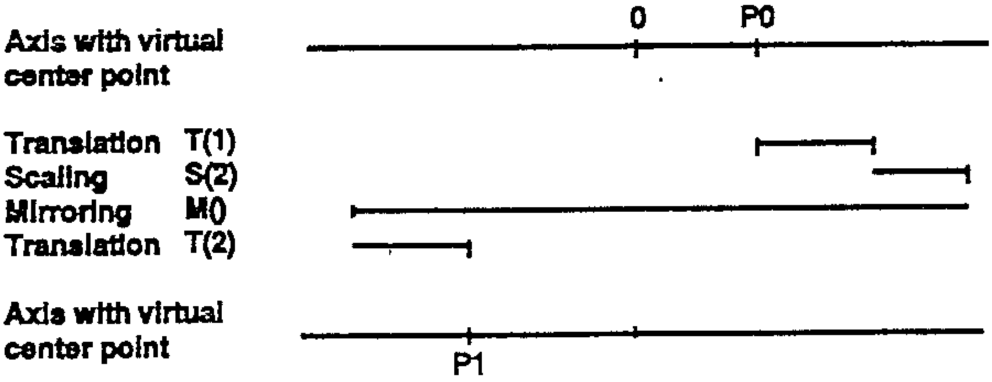


Figure 13 - Sequence of Primitive Axis Value Transformations

5.1.4.3 Coordinate Transformations

A coordinate transformation defines, starting from one coordinate system, the position and orientation of another coordinate system. The transformation from one coordinate system to another should be defined by a rotation (Euler angles) and a vector, in accordance with the conventions established in ISO 841. Objects which may represent pallets, fixtures, parts, or partitions, as defined by this standard, may have offset attributes which refer to coordinate transformations.

An example coordinate transformation is shown in figure 14.

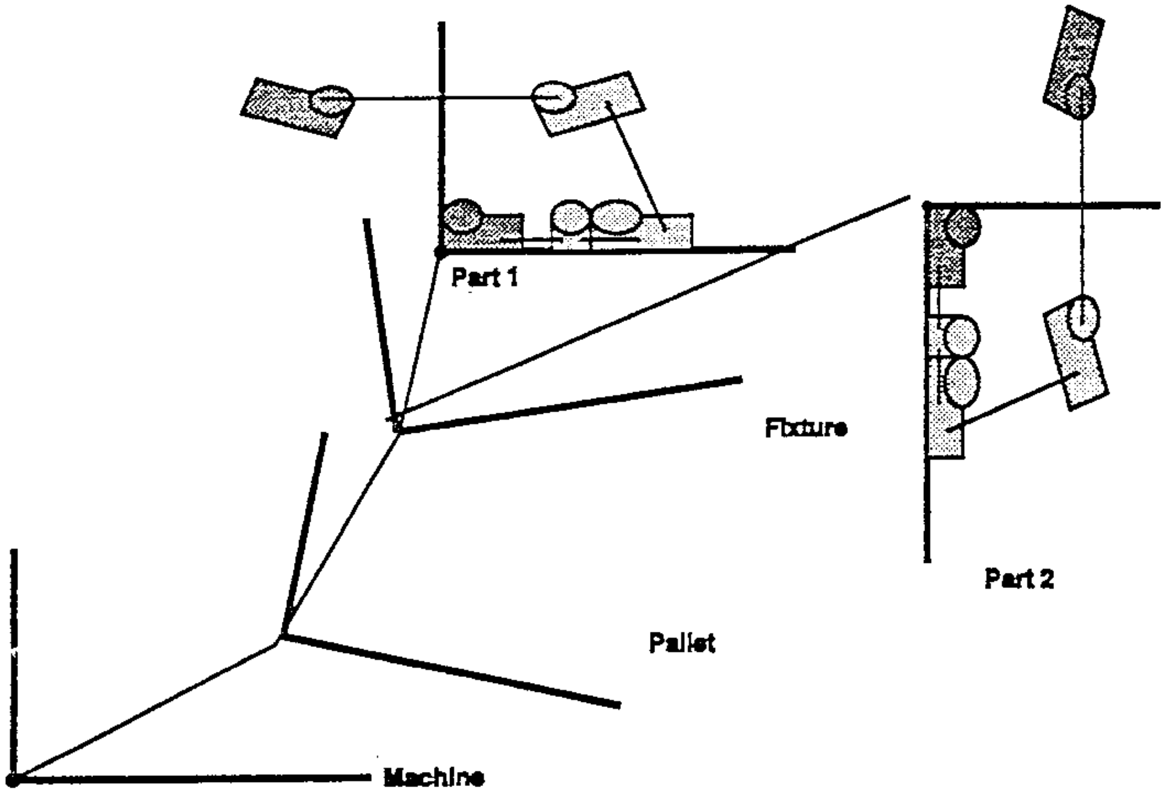


Figure 14 - Coordinate Transformation

5.1.4.4 Active Program Coordinate System

User programs are written for workpieces or parts in terms of their own specific coordinate systems. To execute correctly, sufficient information about such a coordinate system is required within the NC. This information shall be contained in an object known as the "Active Program Coordinate System".

5.1.4.5 Geometry Transformation

Geometry Transformation defines the transformation applied to the coordinates of a point in three dimensional space, to change its reference from the original coordinate system, usually the "program coordinate system", to some target coordinate system, usually the coordinate system of the machining environment.

The effect of geometry transformations is the change of location, orientation, or geometry of the workpiece produced by the part program.

The following defines the primitives used in geometry transformations. The original coordinate system is assumed to be in terms of the coordinates x, y, and z:

Translation:

Translation of point P0 with vector v
primitive T(v): P1 = T(v)*P0 (matrix notation)

Scaling:

Scaling of the x,y,z components of P0 with scale factors s1, s2, s3.
primitive S(s1,s2,s3): P1 = S(s1,s2,s3)*P0 (matrix notation)

Rotation:

Sequence of three rotations (Rotation defined by ISO 841), defined in the following order:
1. Rotation about x axis with angle a
2. Rotation about y axis with angle b
3. Rotation about z axis with angle c

primitive R(a,b,c): P1 = R(a,b,c)*P0 (matrix notation)

Mirroring:

M (xy, yz, xz)

For xy, yz and xz only the two values 0 and 1 are valid.
0 means no mirroring about the specified plane.
1 means mirroring about the specified plane.

Examples:

mirroring about the xy plane:	M(1,0,0)
mirroring about the yz plane:	M(0,1,0)
mirroring about the xz plane:	M(0,0,1)
mirroring about the xy and yz plane:	M(1,1,0)

Mirroring by itself is commutative. Otherwise, any sequence of primitive operations is order dependent. Following is a commonly used notation which reflects the order (see figure 15):

```
P1 = T(1,0,0) * R(10,20,30) * M(1,0,0) * P0
                              ^ first primitive
                ^ second primitive
      ^ third primitive
```

NOTE - When using the mirroring primitive, the shape of a workpiece remains the same. For example, if the program describes a cylinder, then after a mirroring operation the cylinder remains a cylinder. When using the scaling primitive with negative scale factors, the shape changes. Thus, the cylinder program, after scaling with the scale factor -1, will produce a hole in the work piece.

NOTE - This standard does not limit the number of primitive transformations, which shall be implementation specific.

NOTE - Complex geometric transformations may be expressed by a sequence of primitives.

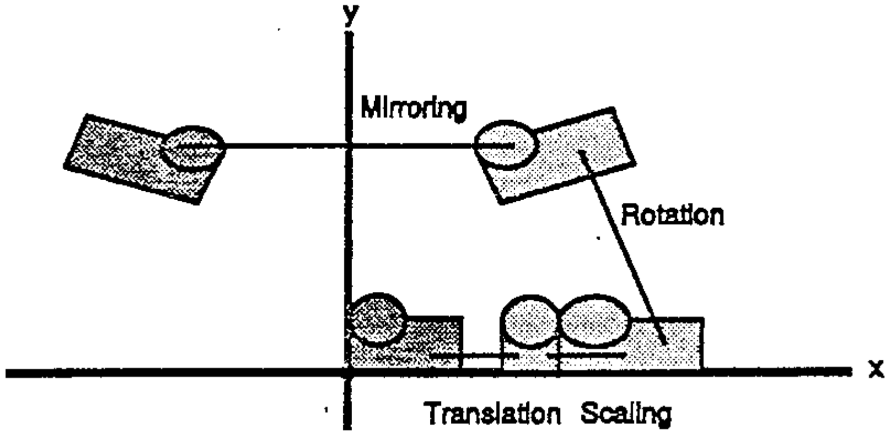


Figure 15 - Sequence of Primitive Geometry Transformations

5.1.5 NC Alarm Model

In order to properly perform its task within a manufacturing environment, an NC shall have the ability to signal faults to an NC-CS user.

NOTE - Such faults may be measuring system faults, control unit temperature problems, control unit battery problems, and so on. Typically, within an NC there exists a list of hundred or more fault conditions, each of which may cause an alarm.
The monitoring and signaling of alarms is represented by the NC Alarm Condition Model. The NC shall include an NC Alarm Condition for every alarm which is to be signaled to an NC-CS user. Alarms are identified by the NC Alarm Condition name. The NC Alarm Condition State represents the current state of monitored alarms.

5.1.5.1 NC Alarm Condition State Model

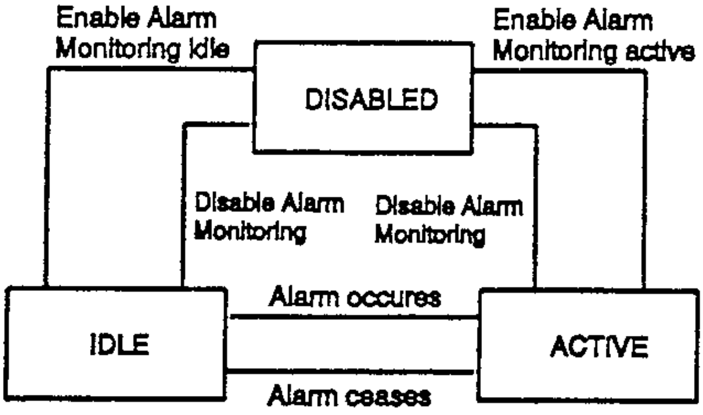


Figure 16 - NC Alarm Condition State Model

5.1.5.1.1 State Description

a) DISABLED - In this state alarm monitoring is disabled.

b) IDLE - This state shall indicate that the monitored alarm is not active or the alarm has ceased to exist.

c) ACTIVE - This state shall indicate that the monitored alarm is active.

5.1.5.1.2 State Transitions

a) Enable Alarm Monitoring idle - This transition enables the alarm monitoring while the monitored alarm is not active.

b) Enable Alarm Monitoring active - This transition enables the alarm monitoring while the monitored alarm is active.

c) Disable Alarm Monitoring - This transition disables the alarm monitoring.

d) Alarm occurs - This transition is associated with the occurrence of the monitored alarm.

e) Alarm ceases - This transition is associated with the discontinuance of the monitored alarm.

5.1.5.2 NC Alarm Notification

If an NC Alarm Condition State changes, an NC Alarm Notification may be sent to the NC-CS user. This NC Alarm Notification shall include the following parameters:

NC Alarm Condition Name - This parameter identifies the NC Alarm Condition to which this NC Alarm Notification applies.

NC Alarm Condition State - This parameter represents the current state of Alarm Monitoring (DISABLED, IDLE, ACTIVE).

NC Alarm Severity - This parameter represents some measure of severity for NC-CS user receiving the NC Alarm Notification.

NC Alarm Time - This parameter contains the time (date and time or a time sequence identifier) at which the transition in the NC alarm condition state was detected.

NC Alarm Acknowledgment Rule - This parameter indicates the level of acknowledgment required for the NC Alarm Notification received (NONE, ACK-ACTIVE, ACK-ALL)

5.1.5.3 Acknowledgment of Alarms

The parameter NC Alarm Acknowledgment Rule directs how the NC-CS user shall acknowledge the receipt of the NC Alarm Notification. The following different levels of acknowledgment are indicated by the NC Alarm Acknowledgment Rule:

a) NONE - Acknowledgments are allowed but not required.

b) ACK-ACTIVE - Acknowledgments shall be required for NC Alarm Notifications specifying the NC Alarm Condition State ACTIVE. Acknowledgments specifying the NC Alarm Condition State IDLE are allowed but not required.

c) ACK-ALL - Acknowledgments shall be required for NC Alarm Notifications specifying the NC Alarm Condition State ACTIVE or IDLE.

NOTE - The Alarm Acknowledgment Rule may be different for every Alarm Condition. For NC Systems not supporting the acknowledgment of alarms this parameter shall have the value NONE.

5.1.6 Switching Model

The switching model is designed to enable an NC-CS user to remotely switch operations within the NC system (such as switch the NC systems from REMOTE CONTROL to LOCAL CONTROL, or to switch an auxiliary function on or off).

5.1.6.1 Switching State Model

Refer to figure 17 for the state diagram of the Switching Model, as observed from the viewpoint of the NC-CS user.

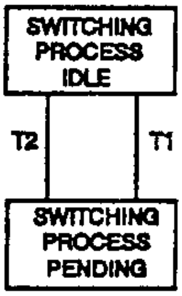


Figure 17 - Switching State Diagram

5.1.6.1.1 State Descriptions

The switching state diagram consists of the two states shown below.

a) SWITCHING PROCESS IDLE - In this state the NC system shall be ready for a command to start the switching process. SWITCHING PROCESS IDLE shall be the default state of the Switching State Model.

b) SWITCHING PROCESS PENDING - This state is reached immediately after receipt of a switching command by the NC-CS user, and the switching process is about to begin.

5.1.6.1.2 State Transitions

The switching state diagram contains the transitions T1 and T2.

a) Transition T1 - This transition occurs upon the receipt of a switching command from the NC-CS user.

b) Transition T2 - This transition occurs after initiating the switching process within the NC system, and the result of the switching process is either "successful" or "not successful".

5.1.7 NC Data Store Model

Most numerical controllers provide data storage, as well as the capability of remote selection, inspection, and/or retrieval of any data required for NC operations. Such data typically consist of user programs and operational parameters, such as tooling, fixturing, process, and other parameters. They are usually generated remotely, and transferred to the NC either via a communication network, or by local means. The Data Store Model defines the organization of the data store, and the manner in which the data may be viewed.

While this standard does not specify any particular data store hierarchy, it does provide for multiple data stores. Such data stores may be within or external to the numerical controller. External store(s) are included in this model so that they may be visible to the NC-CS user.

NOTE - ISO/IEC 9506-4 only defines the model of a logical data storage system from the viewpoint of an NC-CS user, and that the mapping of such model to a real data store is outside the scope of this standard.

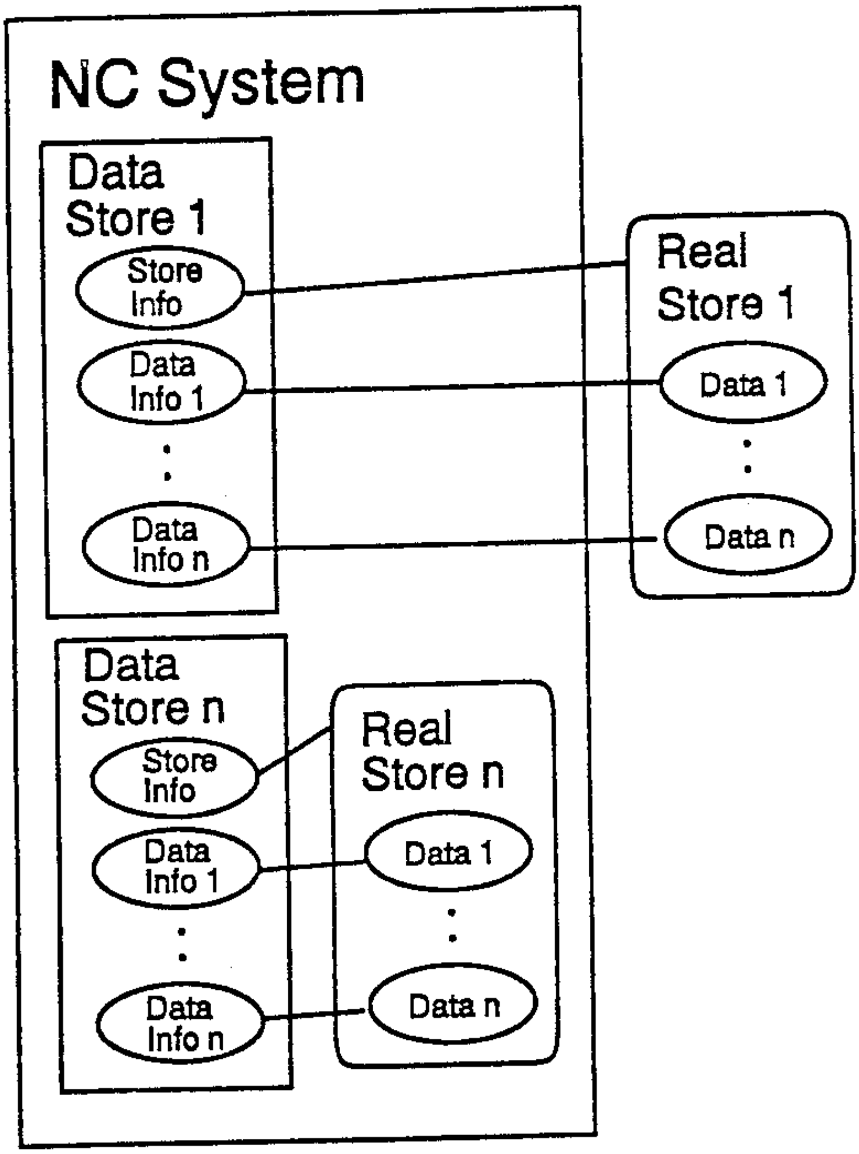


Figure 18 - NC Data Store Model

5.2 NC Specific Functions

This part of ISO/IEC 9506 does not define any functional classes of NC systems. Instead, it defines the following, general NC specific functions, any of which may or may not be supported by a particular NC system. The mapping of these functions to MMS is defined in Chapter 6.

5.2.1 Identify Function

The Identify function enables an NC-CS user to identify a particular NC system and to get information about its capabilities, vendor, revision number, and so on.

5.2.2 Data Transfer

Numerical controllers conforming to this international standard shall be able to transfer data from the NC-CS user to the NC system and back. There are 3 different ways to transfer data to/from an NC system:

a) "block transfer to datastore" - In this way data are exchanged as "bulk" data between the NC-CS user and the NC system (Download and Upload).

b) "block transfer - dynamic" - In this way the NC system executes the data as it is received, with minimal buffering (Dynamic Download).

c) Variable Transfer - In this way data are viewed as variables and the data transfer is done by reading or writing these variables.

The NC-CS user may also direct the NC system to load or store data from a remote fileserver.

Some data, such as tooling data, may be handled either as data blocks or as variables, or both, depending on the complexity of a particular application.

Simple numerical controllers may handle tooling data using block transfer, and more complex systems may use either block transfer or variable transfer, or both. Variable data structures to handle a variety of applications are defined in the normative Annexes to ISO/IEC 9506-4.

5.2.3 NC Data Store Management

The NC Data Store Model provides to the NC-CS user the functionality which addresses the following data management capabilities:

a) Storage Organization -- the names of the NC-CS user programs in the data store(s).

b) Version Control -- the origin and/or version attribute of NC-CS user programs in the data store(s).

c) Memory Usage -- the amount of data store space allocated to NC-CS user programs.

d) Memory Location -- the kind of storage medium represented by the data store.

e) Data Type -- the format of NC-CS user programs in the data store.

f) Protection Schema -- the integrity, or accessibility of the data in the data store.

g) Deletion of data store entries.

5.2.4 Process management

For remote control of the NC system, the following functions which relate to program control are required:

- The ability to select a machine program for execution.
- The ability to start the execution of a Controlling Process. (Remote cycle start)
- The ability to pause and resume a Controlling Process.
- The ability to stop the execution of a Controlling Process.

5.2.5 Interaction with an operator

If there is an need for interaction with an operator, the operator interface device provides the means for the NC-CS user to communicate with the NC system operator.

5.2.6 Alarm Processing

In order to signal faults to the NC-CS user, NC systems should have the capability to:

- mask/unmask alarms
- signal alarms to the NC-CS user
- receive alarm acknowledgments
- provide summary information of active alarms

5.2.7 Switching Operation

An NC system shall provide an NC-CS user with the capability to perform simple switching operations, such as POWER ON, REMOTE CONTROL to LOCAL CONTROL, and many more.

6 NC Application Specific Context Mapping

This clause relates the model of the NC system which has been defined in clause 5, to the abstract model of the Virtual Manufacturing Device (VMD) defined in MMS. For the purpose of inter-operability the uniform application of the abstract concepts in MMS to real systems is essential. While this part of ISO/IEC 9506 may not anticipate every variant of NC System existing or yet to be developed, it provides general guidelines for associating real systems with the abstract model, which should be applicable to any conforming NC system.

6.1 Mapping Of The NC Specific Model To The VMD Object

This clause shows how different elements of NC systems relate to the MMS abstract concepts of VMD, Domains, Program Invocations, and Variables. It is equally applicable to (simple) NC systems which map to a single VMD, as well as those which map to multiple VMDs. The distinction depends largely upon the application.

6.1.1 Mapping Of The NC System To The VMD Model

This clause defines how the NC specific models of clause 5.1 map to the abstract objects defined in part 1 of this International Standard.

Figure 19 shows the major components of an NC VMD and their relation to each other. An NC VMD may include variables, domains, and one or more Program Invocations.

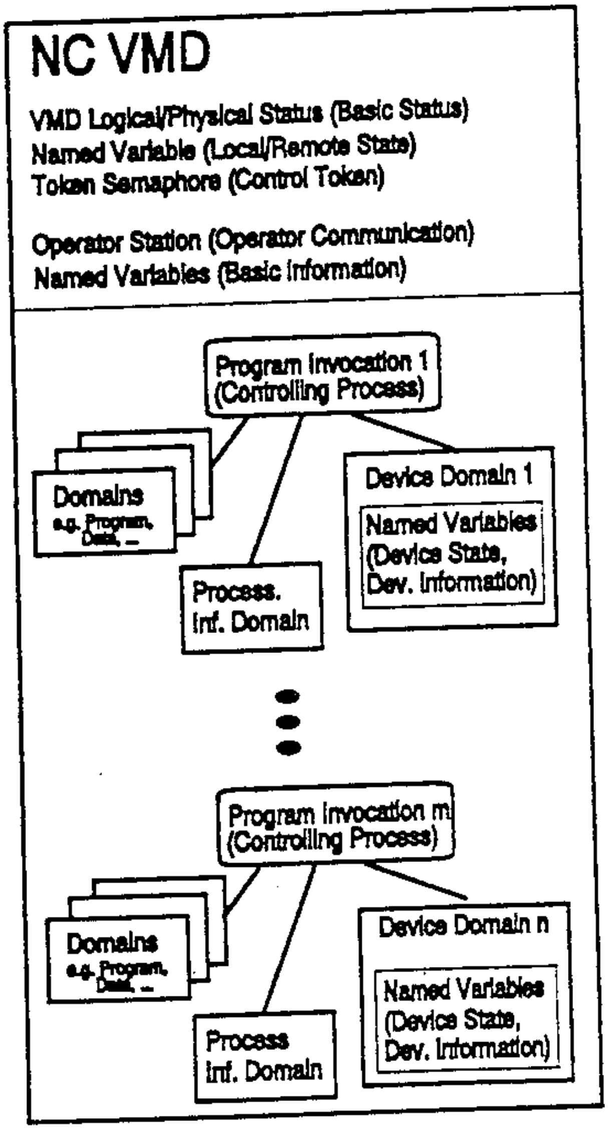

NC VMD
VMD Logical/Physical Status (Basic Status)
Named Variable (Local/Remote State)
Token Semaphore (Control Token)
Operator Station (Operator Communication)
Named Variables (Basic Information)
Program Invocation 1 (Controlling Process)
Domains e.g. Program, Data, ...
Process. Inf. Domain
Device Domain 1
Named Variables (Device State, Dev. Information)
Program Invocation m (Controlling Process)
Domains e.g. Program, Data, ...
Process Inf. Domain
Device Domain n
Named Variables (Device State, Dev. Information)

Figure 19 - NC VMD Model

6.1.1.1 Basic Status

The Basic Status maps to the VMD Logical Status and to the VMD Physical Status.

6.1.1.2 Local/Remote State

The Local/Remote State is modelled by the Switching Model (see clause 6.4.5) and maps to VMD specific Named Variable Objects with the names N_SWP_REMOTE and N_SWC_REMOTE.

6.1.1.3 Control Token

The control token maps to a Token Semaphore Object of the name N_CONTROL.

6.1.1.4 Device Image

A device image maps to a predefined domain object (Device Domain) with domain specific Named Variable objects which represent the physical resources and the state of the Device.

The device state maps to a Device Domain specific Named Variable N_DEV_STATE.

6.1.1.5 User Program

A user program in modelled as a domain object and is linked to a controlling process.

6.1.1.6 User Data

User data are modelled as a domain object and may be linked to a controlling process

6.1.1.7 Controlling Process

The Controlling Process is modelled as a Program Invocation object linked to the domain objects which are germane to the controlled process, and which may include any one or all of the following domain types: program domains, data domains, device domains, and process information domains.

The Controlling Process state maps to the Program Invocation state.

6.1.1.8 NC Information

The Basic Information, the Device Information and the Controlling Process Information are mapped to variables. The Basic Information maps to VMD specific Named Variables, the Device Information maps to device domain specific variables.

The Process Information maps to domain specific variables. The Process Information domain which contains these variables is locally created with creation of the assigned Program Invocation and shall include the name of the Program Invocation.

Instead of accessing different variables separately, the NC-CS user may define Named Variable List objects consisting of the desired variables. This standard provides Variable List objects for the Basic Information (Basic Information List), for the Device Information (Device Information List) and for the Process Information (Process Information List). The use of these objects, however, is optional.

6.1.1.9 NC Alarm Model

The NC Alarm Model maps to the MMS Event Condition and the MMS Event Enrollment

6.1.1.10 NC Data Store

The following object classes are used to provide the functionality of the data store as required by the NC Data Store Model:

- NC Domain Object Class:
    - NC Data Store Domain (N_STORE_...)
- Named Variable Object Class:
    - Data Store Information (N_StoreType,
N_MaxStoreEntries,
N_MaxStoreCapacitiy,
N_CurrentEntries,
N_RemainingEntries,
N_CurrentCapacityUsed,
N_RemainingCapacity)
    - Data Store Entry
- Named Variable Type Object Class:
    - Data Store Entry Type (N_DataStoreEntry)
- Named Variable List Object Class:
    - Data Store Entry List (N_EntryList)

6.1.1.11 Operator Interface Device

The Operator Interface Device maps to the Operator Station object in MMS.
There may be several Operator Station objects be defined in an NC system.

6.1.2 NC VMD

6.1.2.1 NC VMD Object

Object: NC VMD

All MMS defined VMD attributes

Additional Detail - There are no additional attributes defined in this standard.

6.1.2.2 VMD Logical Status and VMD Physical Status

The VMD Logical and VMD Physical Status produce the Basic Status according to the following table:

Table 2 - Status Coherence

Basic Status	VMD Logical Status	VMD Physical Status
NORMAL PRODUCTION LOCAL ACTION REQUIRED	State Changes Allowed any	Operational Needs Commissioning, Inoperable Partially Operational

6.2 Definition Of NC Specific Objects Which Map To Domains

6.2.1 NC Domain Object

Object: NC Domain

All MMS defined Domain attributes

Additional Detail - There are no additional attributes defined in this standard.

6.2.2 Device Domain

Device Domain represents the physical resources of a Device. It is a predefined domain which may be set up by the manufacturer, and which provides a naming space for domain specific Named Variables, such as Device State, Number of Axes, and so on.

6.2.3 Process Information Domain

The Process Information Domain contains the Process Information of the corresponding Controlling Process. This domain is locally created with creation of a Controlling Process and exists only during the lifetime of the same, that is, it is locally deleted with deletion of the Controlling Process.

The Attribute List of Domain References of the associated Program Invocation contains a reference to the Process Information Domain.

6.2.4 Program Domain

User programs usually contain a sequence of part execution steps and are coded in a high level language (such as ISO 6983). Such programs are modelled as domains, which provides the means to upload, download or delete user programs from the NC store, and enables an NC-CS user to assemble several user programs into one program invocation.

6.2.5 Data Domain

Data used by user programs may be modelled as domains or variables in domains, for example:

- Tool Data domain: manufacturer defined tooling information.
- Means of Production Data domain: manufacturer defined means of production information, such as fixture information.

Like program domains these objects may be uploaded, downloaded, or deleted from the NC store, and may be assembled, together with user programs, into a program invocation.

6.2.6 NC Data Store Domain

An NC Data Store domain object shall be created for each logical data store in the NC VMD. Such a domain shall contain certain domain specific objects, namely those which define the NC data store, and those which define domains assigned to the data store. The name of an NC Data Store domain shall begin with "N_STORE".

The creation of an NC Data Store domain is a local matter. This standard does not map a logical data store to a physical data storage device.

The NC Data Store Domain list of subordinate objects includes the following:

- Data Store Information -- corresponds to the "Store Info" element of the NC Data Store model (see figure 18).
- Data Store Entry Type -- defines the content of "Data Info" element of the model.
- Data Store Entries -- corresponds to the "Data Info" element of the model.
- Data Store Entry List -- provides a list of Data Info element names.

The relationship between these objects and the NC Data Store model is shown in figure 20.

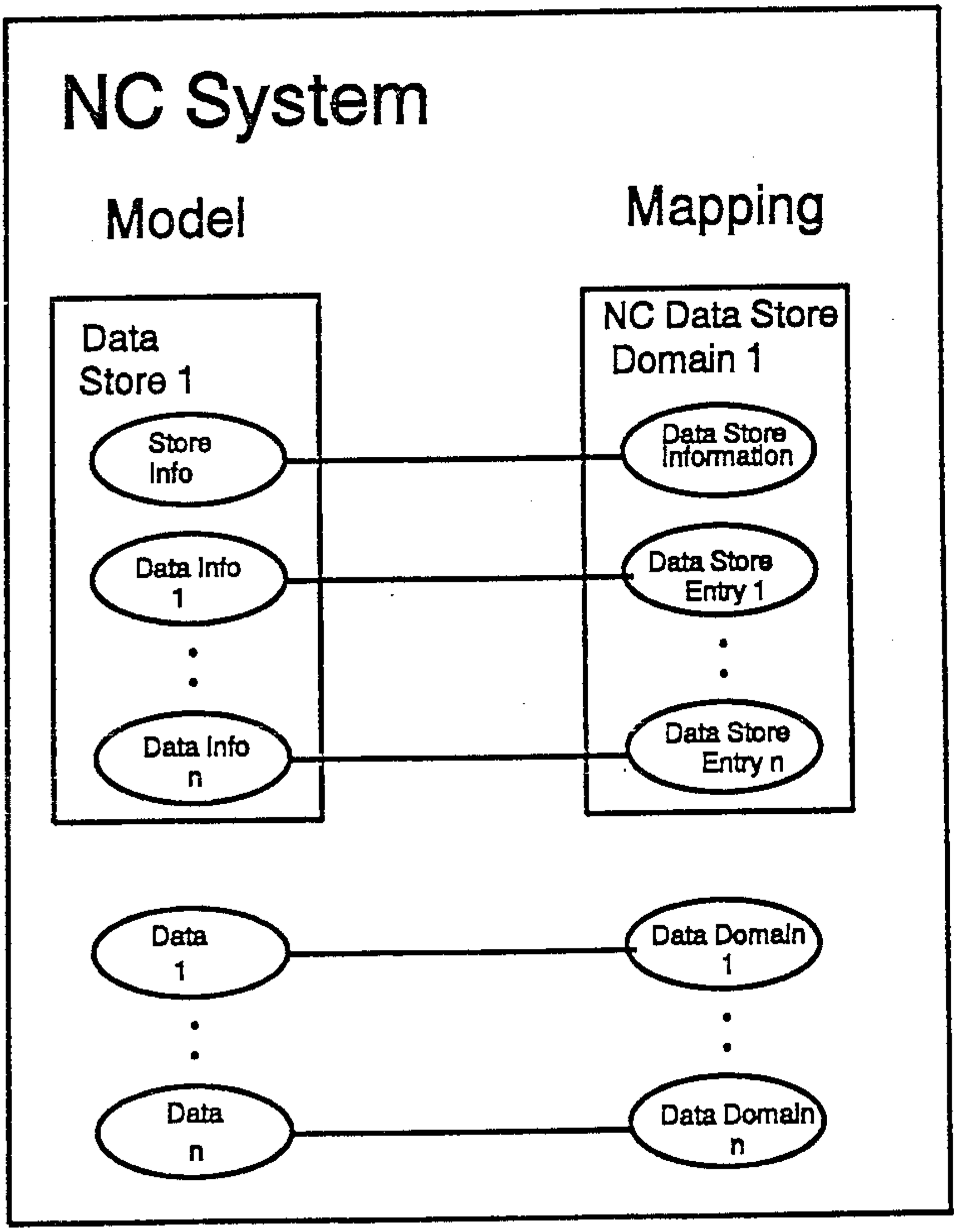


Figure 20 - Data Store Mapping

The NC Data Store is normally not modified directly. Its objects are created and deleted locally as a result of the creation and removal of domains from the NC VMD. How the NC Data Store image is altered, that is, the creation of NC Data Store domains, and the assignment of objects which are NC Data Store domain specific, shall be a local issue. Following are the possible visible side effects of an upload and download of domains in terms of the NC Data Store:

When a Domain state transition to READY occurs, it should be assumed that a Data Store Entry is created in the NC Data Store domain associated with its Domain Content. The attributes of the Data Store Entry should be derived locally.

When a Domain state transition to NON-EXISTENT occurs, it should be assumed that the associated Data Store Entry is removed from the NC Data Store domain.

The Data Store Information Variable values are also assumed to be modified when an assigned Domain state transition to READY or NON-EXISTENT occurs.

This part of ISO/IEC 9506 does not define the client usage of the values or existence of any NC Data Store objects. It is intended for the NC-CS user to use this information to achieve the functionality defined in clause 5 of this part of this standard.

6.2.7 Mapping Of Other Objects To Domains

Various other objects may be mapped to program domains or data domains. They may include, but not be limited to, the following objects:

- Setup Data: Equipment Setup Data
- Statistical Data: Data collected dynamically (system logs, for example)
- Device Origin Setup Domain: The data and process used for 'Homing' the device.
- Probe Data Table: Probe data values
- Diagnostic Data: Diagnostic Programs or Diagnostic Data which may also be mapped to domains.

While these domain objects may be uploaded, or deleted from the NC, they may not be incorporated in Program Invocations.

6.3 Definition Of NC Specific Objects Which Map To Program Invocation

6.3.1 NC Program Invocation Object

Object: NC Program Invocation
All MMS defined Program Invocation Attributes
Additional Detail - There are no additional attributes defined in this standard.

6.3.2 Controlling Process Program Invocation

The Controlling Process State Model maps directly to the NC Program Invocation State Model. Transitions of the Controlling Process state may be the result of a Program Invocation service or of local action.

Figure 21 shows the transitions of the Controlling Process states which may occur as a positive result of Program Invocation services. The states and transitions of the Controlling Process map directly to the MMS Program Invocations states defined in ISO/IEC 9506, Part 1. To keep the diagram simple it does not show the transient states (STARTING, STOPPING, RESUMING, RESETTING). These transient states shall be used exactly as defined in ISO/IEC 9506, Part 1.

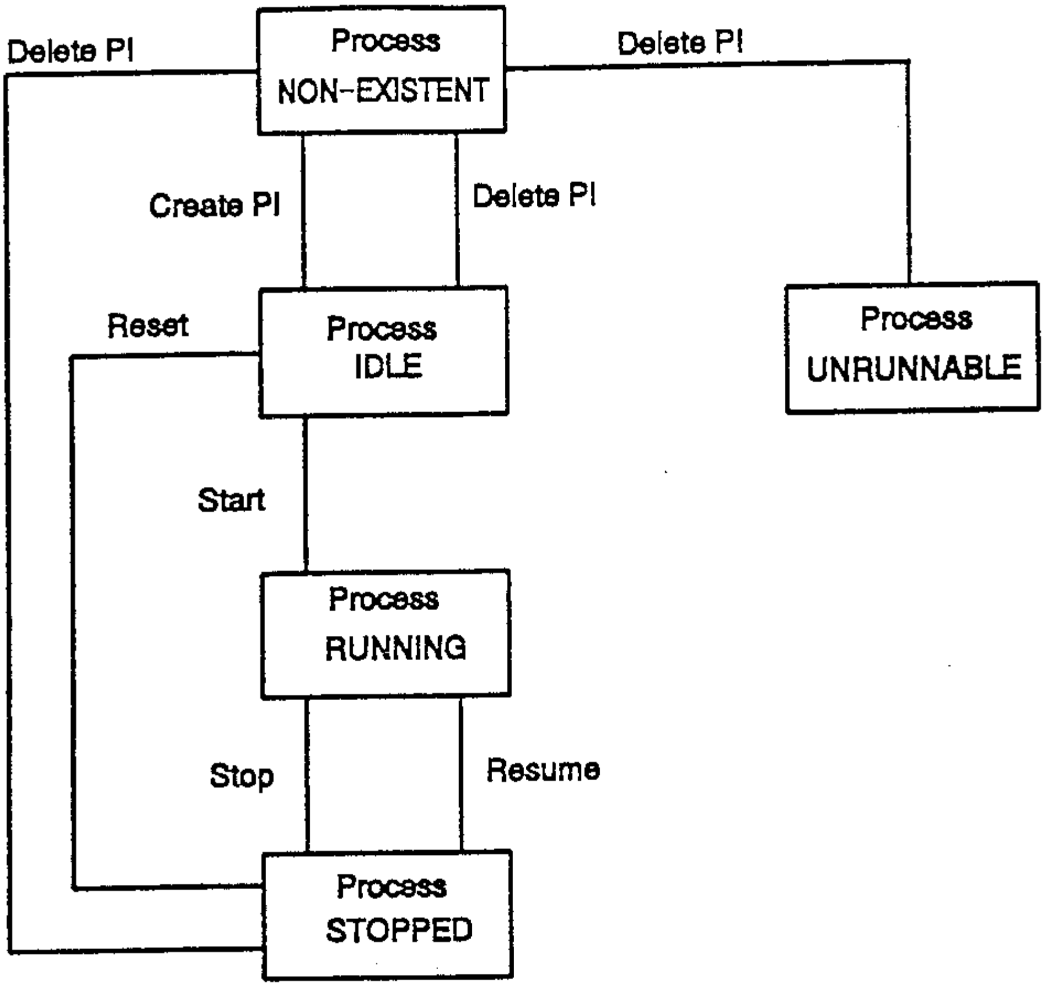


Figure 21 - Transitions of the Controlling Process state as a result of Program Invocation services

In Figure 19 one device domain is linked to one Controlling Process. This configuration will be the normal case. But this standard does not restrict the VMD model to this one to one relationship.

Thus, an NC-CS user may create a second user program linking other programs and data to the same Device Domain while the first program is still executing.

Figure 22 shows an example of an NC VMD with two Program Invocations linked to a single Device Domain.

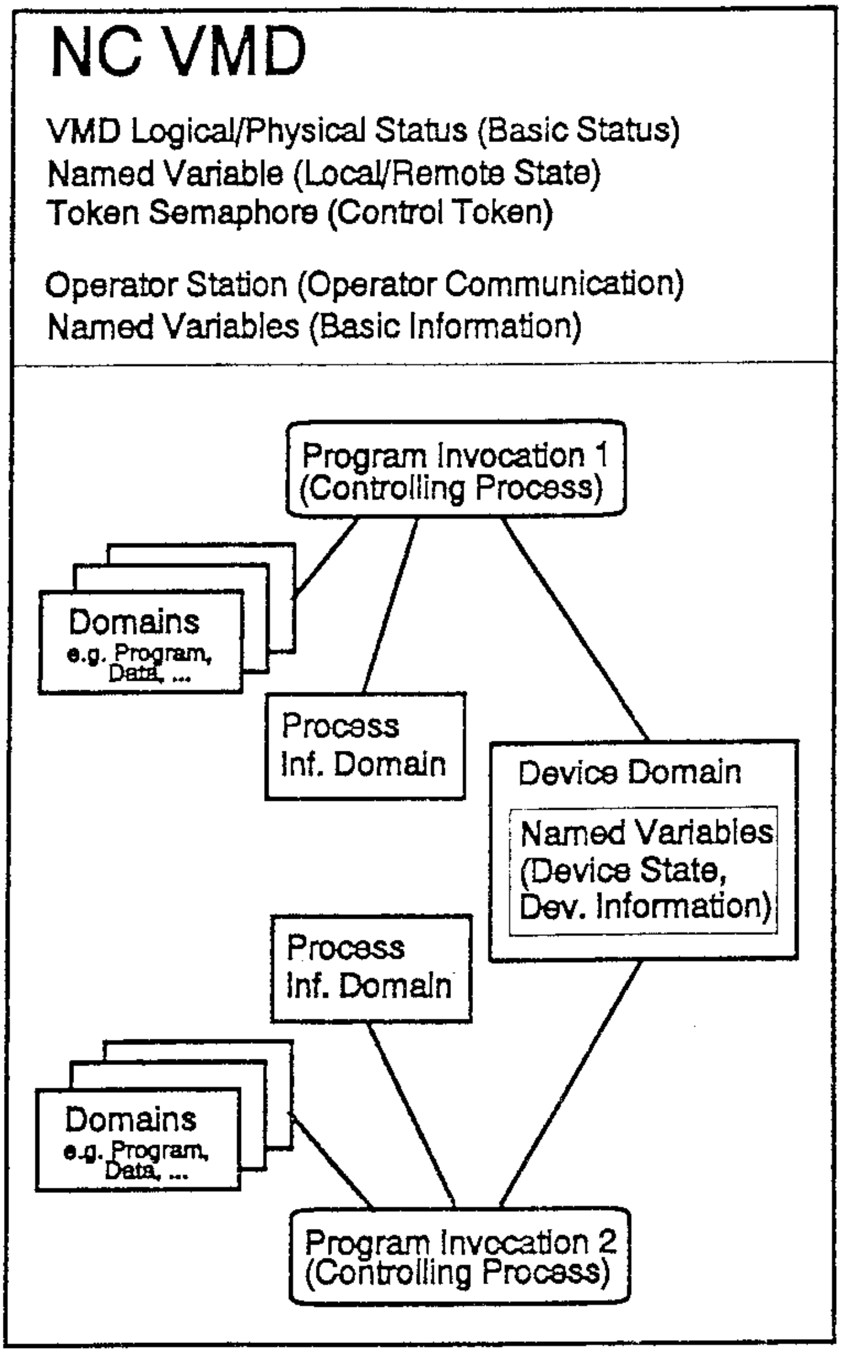

NC VMD
VMD Logical/Physical Status (Basic Status)
Named Variable (Local/Remote State)
Token Semaphore (Control Token)
Operator Station (Operator Communication)
Named Variables (Basic Information)
Program Invocation 1 (Controlling Process)
Domains e.g. Program, Data, ...
Process Inf. Domain
Device Domain
Named Variables (Device State, Dev. Information)
Process Inf. Domain
Domains e.g. Program, Data, ...
Program Invocation 2 (Controlling Process)

Figure 22 - NC VMD Model (2 Program Invocations, 1 Device Domain)

There may also be situations which require that several devices be linked to one Controlling Process. Such a process is the Device Origin Setup program which may require access to more than one device. Figure 23 shows the example of an NC VMD with two Device Domains linked to one Program Invocation.

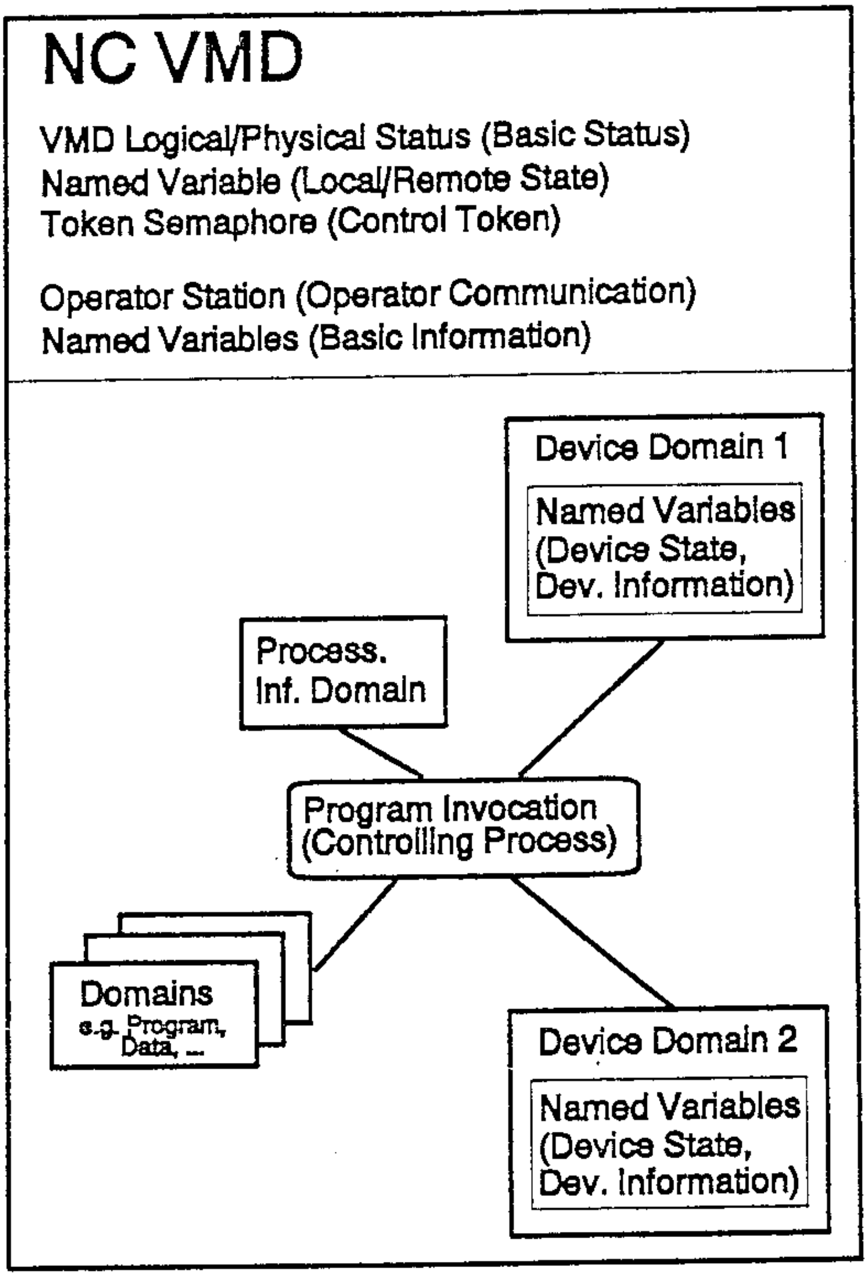

NC VMD
VMD Logical/Physical Status (Basic Status)
Named Variable (Local/Remote State)
Token Semaphore (Control Token)
Operator Station (Operator Communication)
Named Variables (Basic Information)
Device Domain 1
Named Variables (Device State, Dev. Information)
Process. Inf. Domain
Program Invocation (Controlling Process)
Domains e.g. Program, Data, ...
Device Domain 2
Named Variables (Device State, Dev. Information)

Figure 23 - NC VMD Model (1 Program Invocation, 2 Device Domains)

6.4 Definition Of NC Specific Objects Which Map To Other MMS Abstract Objects

6.4.1 Definition Of NC Specific Objects Which Map To Named Variables

The MMS Variable Access services provide facilities by which typed variables of an MMS VMD may be accessed. MMS defines the Unnamed Variable object, the Named Variable object, and the Scattered Access object.

Of these MMS variable objects, only the support of the Named Variable object is required for conformance with the NC VMD defined in this part of ISO/IEC 9506.

6.4.1.1 NC Named Variable Object

Object: NC Named Variable
All MMS defined Named Variable Attributes

6.4.1.2 Device State

The Device State maps to a Device Domain specific Named Variable. The name and structure of this variable is defined in clause 8.3.

6.4.1.3 Basic Information

The Basic Information maps to VMD specific Named Variables. A name prefix and the structure of these variables is defined in clause 8.3.

6.4.1.4 Device Information

The Device Information maps to Device Domain specific Named Variables. Name prefix and structure of these variables are defined in clause 8.3.

6.4.1.5 Process Information

The Process Information maps to domain specific Named Variables. The Process Information domain which contains these variables only exists during the life of its corresponding Controlling Process, and so do these Process Information variables.

Name prefix and structure of Named Variables Lists representing Controlling Process Information are defined in clause 8.5.

6.4.1.6 Data Store Information

Data Store Information shall be mapped to the following Named Variable Objects which are contained in the NC Data Store Domain:

N_StoreType – shall identify the type or other characteristic of the storage medium.

N_MaxStoreEntries – shall specify the maximum number of data store entry objects this data store may hold.

N_MaxStoreCapacity – shall specify the maximum octet count of data which this data store may hold.

N_CurrentEntries – shall specify the current number of data store entries contained in this data store.

N_RemainingEntries – shall specify the number of available entries which are unused.

N_CurrentCapacityUsed – shall specify the octet count of the space used by data currently contained in this data store.

N_RemainingCapacity – shall contain the octet count of the space which is currently unused in this data store.

6.4.1.7 Data Store Entry

Data store entries shall be mapped to Named Variable objects. Name prefix and structure of these variables are defined in clause 8.3. Part of the identification of a data store entry shall be the name of the NC Domain object which the entry represents.

6.4.1.8 Offset model

The offset model shall be mapped to the following Named Variable Objects:

- N_ZeroAxes –contains the zero offset values
- N_AxisValTrans – contains the axis transformation values
- N_Machine_CS – describes the machine coordinate system
- N_ActProg_CS – identifies the active program coordinate system
- N_GeometryTrans -- contains the geometry transformation values.

6.4.2 Definition Of NC Specific Objects Which Map To Named Variable List Objects

The MMS Named Variable List object provides for the assignment of one name for a group of (more or less) independent MMS Variables which may be accessed together. Thus, the NC-CS user may read the Device Information, by accessing its variable list name, without the need to separately access the variables contained therein.

6.4.2.1 NC Named Variable List Object

Object: Named Variable List
All MMS defined Named Variable List Attributes

6.4.2.2 Basic Information List

The Basic Information List shall contain one or more of the named variables representing Basic Information. Naming of the Basic Information List is defined in clause 8.5.

6.4.2.3 Device Information List

The Device Information List shall contain one or more of the named variables representing Device Information. Naming of the Device Information List is defined in clause 8.5.

6.4.2.4 Process Information List

The Process Information List shall contain one or more of the named variables representing Process Information. Naming of the Process Information List is defined in clause 8.5.

6.4.2.5 Data Store Entry List

The Data Store Entry List shall contain one or more of the named variables representing data store entries. Naming and structure of the Data Store Entry List are defined in clause 8.5.

The unique identifier of the Data Store Entry List shall be N_EntryList.

6.4.3 Definitions Of NC Specific Objects Which Map To Named Type Objects

Object: NC Named Type Objects
All MMS defined Named Type Object Attributes

NOTE - This standard does not standardize any services for NC specific named type objects.

6.4.3.1 Data Store Entry Type

There shall be only one Named Type Variable in the NC Data Store domain. This use of a single named type variable shall allow implementers to extend the scope of the data store model to include attributes not defined in this standard.

The only restriction on implementers is that they shall include the following data type attributes with their new data types. If this restriction is followed, the conforming client will be able to retrieve the attributes of the Data Store Entry type, and to access any Data Store Entry object via the read variable service using the standardized attributes.

Naming and structure of the Data Store Entry Type are defined in clause 8.6.
Its unique identifier shall be N_DataStoreEntry.

The attributes of the Data Store Entry Type shall include:

a) Data Name -- a visible string equal to the name of a Domain object whose content is assigned to this data store.

b) Data Type -- a visible string identifying the format of the domain content as it would be uploaded or downloaded. This string shall be equal to "Non-Coded" if the non-coded choice of loadData should be used, and equal to the name of the OBJECT IDENTIFIER of the syntax used if the coded choice of load Data should be used.

c) Data Size -- a number representing the count of octets assigned to the domain which this entry represents, from the N_CurrentCapacityUsed attribute of the data store domain.

d) Creation Date -- a value representing the time of creation of the domain which this entry represents.

e) Last Edit -- a value representing the time of the last modification of the content of the domain which this entry represents.

f) Protection Mode -- a visible string representing the access level or privilege required to access the content of the domain which this entry represents.
Default = empty.

g) Data Source -- a visible string representing the location from which the content of the domain which this entry represents, was obtained.
Default = empty.

6.4.4 Definition Of NC Specific Objects Which Map To Event Condition, Event Action And Event Enrollment Objects

6.4.4.1 NC Alarm Model

The NC Alarm Model maps to the MMS Event Model. The NC Alarm Condition maps to an Event Condition object linked to one or more Event Enrollment objects. The sending of an NC Alarm Notification maps to the MMS EventNotification Service. Acknowledgment of an NC Alarm Notification maps to the MMS AcknowledgeEventNotification Service. If the NC supports the request of summary information about the current status of NC Alarm Conditions, then it shall use the MMS GetAlarmSummary Service for this. NC-CS users may mask or unmask alarms by using the MMS AlterEventConditionMonitoring service. It should also be possible to mask or unmask alarms by local action.

6.4.4.1.1 NC Event Condition Object

Object: NC Event Condition
All MMS defined Event Condition Attributes
Additional Detail - No additional attributes are defined in this standard.

NC Event Conditions typically are predefined. This standard specifies that the MMS Event Condition Attributes be restricted to the following:

Attribute: Event Condition Name = NC Alarm Condition Name
Attribute: Event Condition Class = MONITORED
Attribute: State = NC Alarm Condition State
Attribute: Severity = NC Alarm Severity

6.4.4.1.2 NC Event Enrollment Object

Object: NC Event Enrollment
All MMS defined Event Enrollment Attributes
Additional Detail - No additional attributes are defined in this standard.

NC Event Enrollments typically are created in the connection establishment phase by local means. This standard specifies that the MMS Event Enrollment Attributes be restricted to the following:

Attribute: Enrollment Class = NOTIFICATION
Attribute: Alarm Acknowledgment Rule = NC Alarm Acknowledgment Rule

6.4.5 Mapping of the Switching Model to MMS Objects

This clause defines the mapping of the general switching operation model developed in clause 5.1.5 to the abstract objects and services defined in ISO/IEC 9506-1 and ISO/IEC 9506-2.

The switching model is based on two named variable objects of the type BOOLEAN. The first variable, "Switch_Command", acts as a switching mechanism trigger. The second variable, "Switch_Position", is used to indicate the actual position of the switch or the resulting position of the switching operation. Further information about these two variables may be found in clause 8.3.

NOTE - The switching model defined in this part of ISO/IEC 9506 shall only be used for simple switching operations. Complex switching operations shall be based on program invocations.

To initiate a switching operation the enrolled NC-CS user shall issue a write.request to set the value of the variable Switch_Command either TRUE or FALSE, depending on whether the desired end result is to switch "ON" or "OFF" (see figure 24).

Switch	Switch_Command
ON	TRUE
OFF	FALSE

Figure 24 - Relation between requested switching and value of Switch_Command

The receipt of a write.indication witch_Command = TRUE or FALSE causes a state transition from the SWITCHING PROCESS IDLE state to the unnamed intermediate state (see figure 25).

Depending on whether the variable write operation has succeeded or failed a write.response (+) or a write.response (-) should be issued by the NC system.

A successful variable write operation shall cause a state transition from the intermediate to the SWITCHING PROCESS PENDING state.

A failed variable write operation shall cause a state transition back from the intermediate to the SWITCHING PROCESS IDLE state.

While in the SWITCHING PROCESS PENDING state, it shall be a local matter for the NC system to actually execute the switching command.

An InformationReport.request shall inform the enrolled NC-CS user about the state of the variable Switch_Position.

The transmission of the InformationReport witch_Position = TRUE or FALSE shall cause a transition from the SWITCHING PROCESS PENDING to the SWITCHING PROCESS IDLE state.

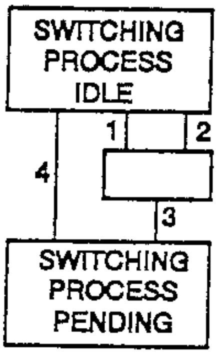


Figure 25 - Switching Process State Diagram

The transitions of the state diagram are as follows:

1 - Write.indication

2 - Write.response (-) or Write.response (+) Success = false

3 - Write.response (+) Success = true

4 - InformationReport.request

NOTES -

(1) On occasion the NC-CS user may attempt set the switch to its current position. In this case, the switching command shall be accepted and the "switching process" executed without changing the position of the switch.

(2) It is a local issue of the NC system to ensure that during a pending switching operation no additional switching order is accepted. The NC should refuse the write access of the Switch_Command variable, in this case.

6.5 Definition Of New MMS Abstract Objects To Support Other NC Specific Objects

There are no new MMS abstract objects defined in this standard.

7 Services

7.1 Use of the MMS services

The abstract syntax as defined herein are part of the ASN.1 module "ISO--9506-NCCS-1".

```
ISO-9506-MMS-NCCS-1
{iso standard 9506 part(4) mms-nccs-module-version1(1)}
 DEFINITIONS ::= BEGIN
 IMPORTS  MMSpdu, ServiceSupportOptions, ParameterSupportOptions, integer16
 FROM MMS-General-Module-1
{iso standard 9506 part(2) mms-general-module-version1(2)};
```

```
CS-Status-Request ::= NULL
CS-Input-Request ::= NULL
CS-Output-Request ::= NULL
CS-InitiateDownloadSequence-Request ::= NULL
CS-DownloadSegment-Request ::= NULL
CS-TerminateDownloadSequence-Request ::= NULL
CS-InitiateUploadSequence-Request ::= NULL
CS-UploadSegment-Request ::= NULL
CS-TerminateUploadSequence-Request ::= NULL
CS-RequestDomainDownload-Request ::= NULL
CS-RequestDomainUpload-Request ::= NULL
CS-LoadDomainContent-Request ::= NULL
CS-StoreDomainContent-Request ::= NULL
CS-DeleteDomain-Request ::= NULL
CS-GetDomainAttributes-Request ::= NULL
CS-CreateProgramInvocation-Request ::= NULL
CS-DeleteProgramInvocation-Request ::= NULL
CS-Start-Request ::= NULL
CS-Stop-Request ::= NULL
CS-Resume-Request ::= NULL
CS-Reset-Request ::= NULL
CS-Kill-Request ::= NULL
CS-GetProgramInvocationAttributes-Request ::= NULL
CS-DefineEventCondition-Request ::= NULL
CS-DeleteEventCondition-Request ::= NULL
CS-GetEventConditionAttributes-Request ::= NULL
CS-ReportEventConditionStatus-Request ::= NULL
CS-AlterEventConditionMonitoring-Request ::= NULL
CS-TriggerEvent-Request ::= NULL
CS-DefineEventAction-Request ::= NULL
CS-DeleteEventAction-Request ::= NULL
CS-GetEventActionAttributes-Request ::= NULL
CS-ReportEventActionStatus-Request ::= NULL
CS-DefineEventEnrollment-Request ::= NULL
CS-DeleteEventEnrollment-Request ::= NULL
CS-AlterEventEnrollment-Request ::= NULL
CS-ReportEventEnrollmentStatus-Request ::= NULL
CS-GetEventEnrollmentAttributes-Request ::= NULL
CS-AcknowledgeEventNotification-Request ::= NULL
CS-GetAlarmSummary-Request ::= NULL
CS-GetAlarmEnrollmentSummary-Request ::= NULL
CS-ReadJournal-Request ::= NULL
CS-WriteJournal-Request ::= NULL
CS-InitializeJournal-Request ::= NULL
CS-ReportJournalStatus-Request ::= NULL
CS-CreateJournal-Request ::= NULL
CS-DeleteJournal-Request ::= NULL
CS-GetCapabilityList-Request ::= NULL
CS-Status-Response ::= NULL
CS-Input-Response ::= NULL
CS-Output-Response ::= NULL
CS-InitiateDownloadSequence-Response ::= NULL
CS-DownloadSegment-Response ::= NULL
CS-TerminateDownloadSequence-Response ::= NULL
CS-InitiateUploadSequence-Response ::= NULL
CS-UploadSegment-Response ::= NULL
CS-TerminateUploadSequence-Response ::= NULL
CS-RequestDomainDownload-Response ::= NULL
CS-RequestDomainUpload-Response ::= NULL
CS-LoadDomainContent-Response ::= NULL
CS-StoreDomainContent-Response ::= NULL
CS-DeleteDomain-Response ::= NULL
CS-GetDomainAttributes-Response ::= NULL
CS-CreateProgramInvocation-Response ::= NULL
```

```
CS-DeleteProgramInvocation-Response ::= NULL
CS-Start-Response ::= NULL
CS-Stop-Response ::= NULL
CS-Resume-Response ::= NULL
CS-Reset-Response ::= NULL
CS-Kill-Response ::= NULL
CS-GetProgramInvocationAttributes-Response ::= NULL
CS-DefineEventCondition-Response ::= NULL
CS-DeleteEventCondition-Response ::= NULL
CS-GetEventConditionAttributes-Response ::= NULL
CS-ReportEventConditionStatus-Response ::= NULL
CS-AlterEventConditionMonitoring-Response ::= NULL
CS-TriggerEvent-Response ::= NULL
CS-DefineEventAction-Response ::= NULL
CS-DeleteEventAction-Response ::= NULL
CS-GetEventActionAttributes-Response ::= NULL
CS-ReportEventActionStatus-Response ::= NULL
CS-DefineEventEnrollment-Response ::= NULL
CS-DeleteEventEnrollment-Response ::= NULL
CS-AlterEventEnrollment-Response ::= NULL
CS-ReportEventEnrollmentStatus-Response ::= NULL
CS-GetEventEnrollmentAttributes-Response ::= NULL
CS-AcknowledgeEventNotification-Response ::= NULL
CS-GetAlarmSummary-Response ::= NULL
CS-GetAlarmEnrollmentSummary-Response ::= NULL
CS-ReadJournal-Response ::= NULL
CS-WriteJournal-Response ::= NULL
CS-InitializeJournal-Response ::= NULL
CS-ReportJournalStatus-Response ::= NULL
CS-CreateJournal-Response ::= NULL
CS-DeleteJournal-Response ::= NULL
CS-GetCapabilityList-Response ::= NULL
CS-UnsolicitedStatus ::= NULL
CS-EventNotification ::= NULL
CsAdditionalObjectClasses ::= NULL
EN-Additional-Detail ::= NULL
EE-Additional-Detail ::= NULL
JOU-Additional-Detail ::= NULL
AS-Filter ::= NULL
```

7.2 Definition and use of Application-specific Services

There are no NC specific Services defined in this standard, other than the NC specific Initiate service.

```
AdditionalService-Request ::= NULL
AdditionalService-Response ::= NULL
AdditionalService-Error ::= NULL
AdditionalUnconfirmedService ::= NULL
```

7.3 The Initiate Service and Protocol

Two related NC-specific parameters are defined by this part of ISO/IEC 9506 for the MMS Initiate service. For the MMS Initiate-RequestPDU this standard defines the CSRequestDetail parameter, and for the MMS Initiate-ResponsePDU, the CSResponseDetail parameter.

7.3.1 Definition of NC-specific Initiate Service

Extensions

For this service the following additional parameters are required to establish the NC context, and identify the functionality to be used in that context.

Table 3 - **NC-specific CS parameters for Initiate**

Parameter Name	Req	Ind	Rsp	Cnf
CS request detail				
Proposed Version Number	M	M		
Proposed Parameter CBB	M	M		
Services Supported Calling	M	M		
CS response detail				
Negotiated Version Number			M	M (=)
Negotiated Parameter CBB			M	M (=)
Services Supported Called			M	M

Proposed version number - is the highest version number which the requester supports

Negotiated version number - is the version number which shall be used during the association.

Proposed Parameter CBB - is the set of parameter building blocks which the calling MMS-user shall support in the NC context. The value in the indication primitive shall be the intersection of the set requested, and the set supported by the MMS-provider in NC context.

Services Supported calling - is the set of services which the calling MMS-user shall support in the NC context. The value in the indication primitive shall be the intersection of the set in the request primitive, and the set supported by the MMS-provider.

Services Supported called - is the set of services which the called MMS-user shall support in the NC context. The value in the confirm primitive shall be the intersection of the set in the response primitive, and the set supported by the MMS-provider.

The definition of "supported" shall be as defined in MMS.

7.3.2 Definition of NC-specific Initiate Protocol

```
InitRequestDetail ::= SEQUENCE {
   proposedVersionNumber          [0] IMPLICIT Integer16,
   proposedParameterCBB           [1] IMPLICIT ParameterSupportOptions,
   servicesSupportedCalling       [2] IMPLICIT ServiceSupportOptions
   }
InitResponseDetail ::= SEQUENCE {
   negotiatedVersionNumber        [0]  IMPLICIT Integer16,
   negotiatedParameterCBB         [1] IMPLICIT ParameterSupportOptions,
   servicesSupportedCalled        [2] IMPLICIT ServiceSupportOptions
   }
```

7.4 End of Module

The following END statement closes the module, opened by begin in section 7.1

END

8 Standardized NC Specific Objects

NC specific MMS objects are standardized in this clause by standardizing the name attributes of these objects. All names and name prefixes which are standardized in this part of ISO/IEC 9506 shall begin with a two character sequence containing a capital letter "N" in the first character position of the name and an underscore character, "_", in the second character position of the name.

While not all names standardized herein may be supported by all implementations, those which are supported shall be listed in the appropriate PICS entry (see the conformance clause).

If an implementation uses an object whose semantics matches the semantics of any standardized NC specific object defined herein, then such implementation shall use the standardized object, rather than a privately defined object with a non-standardized name. Conversely, implementations may not use any standardized name in any way other than defined in this part of ISO/IEC 9506.

8.1 Domain Objects

The name of each of the standardized NC specific domain objects defined in the following subclauses, shall be constructed in the way specified.

8.1.1 Device Domain

The name of any device domain shall begin with the six character prefix "N_DEV_", complemented by a user or implementer defined device identifier.

Object: Domain

Key Attribute: Domain Name = "N_DEV_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable = FALSE
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.2 Program Domain

The name of any program domain shall begin with the six character prefix "N_PRG_", complemented by a user or implementer defined machine program identifier.

Object: Domain

Key Attribute: Domain Name = "N_PRG_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.3 Process Information Domain

The name of any process information domain shall begin with the six character prefix "N_PID_", complemented by its associated program invocation identifier.

Object: Domain

Key Attribute: Domain Name = "N_PID_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.4 Tool Data Domain

The name of any tool data domain shall begin with the six character prefix "N_TLD_", complemented by the user or implementer defined means of production data table identifier.

Object: Domain

Key Attribute: Domain Name = "N_TLD_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.5 Means Of Production Data Domain

The name of any means of production data domain shall begin with the six character prefix "N_MOP_", complemented by a user or implementer defined identifier.

Object: Domain

Key Attribute: Domain Name = "N_MOP_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.6 Setup Data

The name of any setup data domain shall begin with the six character prefix "N_SET_", complemented by a user or implementer defined machine setup data identifier.

Object: Domain

Key Attribute: Domain Name = "N_SET_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.7 Statistical Data

The name of any statistical data domain shall begin with the six character prefix "N_SDD_", complemented by a user or implementer defined statistical data identifier.

Object: Domain

Key Attribute: Domain Name = "N_SDD_..."

Attribute: List of Capabilities
Attribute: State
Attribute: MMS Deletable
Attribute: Sharable
Attribute: Domain Content
Attribute: List of Subordinate Objects
Attribute: List of Program Invocation References
Attribute: Upload in Progress

8.1.8 Probe Data Table

The name of any probe data table shall begin with the six character prefix "N_PRB_" complemented by a user or implementer defined probe data table identifier.

Object: Domain

Key Attribute: Domain Name = "N_PRB_..."

Attribute:	List of Capabilities
Attribute:	State
Attribute:	MMS Deletable
Attribute:	Sharable
Attribute:	Domain Content
Attribute:	List of Subordinate Objects
Attribute:	List of Program Invocation References
Attribute:	Upload in Progress

8.1.9 Diagnostic Data

The name of any diagnostic data domain shall begin with the six character prefix "N_DGN_" complemented by a user or implementer defined diagnostic data identifier.

Object: Domain

Key Attribute: Domain Name = "N_DGN_..."

Attribute:	List of Capabilities
Attribute:	State
Attribute:	MMS Deletable
Attribute:	Sharable
Attribute:	Domain Content
Attribute:	List of Subordinate Objects
Attribute:	List of Program Invocation References
Attribute:	Upload in Progress

8.1.10 Device Origin Setup Domain

The standardized name "N_DOS" shall be used to identify the domain object containing the device origin setup procedure.

Object: Domain

Key Attribute: Domain Name = "N_DOS"

Attribute:	List of Capabilities
Attribute:	State
Attribute:	MMS Deletable = FALSE
Attribute:	Sharable
Attribute:	Domain Content
Attribute:	List of Subordinate Objects
Attribute:	List of Program Invocation References = "N_DOS"
Attribute:	Upload in Progress

8.1.11 NC Data Store Domain

The name of any NC Data Store domain shall contain the characters "N_STORE_", which may constitute either the full name of the data store domain or a prefix to be complemented by an underscore and a data store identifier, at the option of the user or implementer.

Object: Domain

Key Attribute: Domain Name = "N_STORE_..."

Attribute:	List of capabilities	
Attribute:	State	= READY
Attribute:	MMS Deletable	= FALSE
Attribute:	Sharable	= FALSE
Attribute:	Domain Content	
Attribute:	List of Subordinate Objects	= {
	N_StoreType,	
	N_MaxStoreEntries,	
	N_MaxStoreCapacity,	
	N_CurrentEntries,	
	N_RemainingEntries,	
	N_CurrentCapacityUsed,	
	N_RemainingCapacity,	
	N_DataStoreEntry,	– NOTE 1
	N_EntryList,	
	data store entries	– NOTE 2
	}	
Attribute:	List of Program Invocation References	= {} – NOTE 3
Attribute:	Upload in Progress	

Notes -

(1) Implementers may substitute a different named type object which includes the attributes of N_DataStoreEntry.

(2) Data store entries may also be of an extended type defined by a substitute named type variable.

(3) The List of program invocation references shall be an empty list.

8.2 Program Invocation Objects

Each of the standardized NC specific objects defined in the following subclauses, which are mapped to Program Invocation objects, shall be given the name specified.

8.2.1 Program Invocation Object N_DOS

The Controlling Process Program Invocation object with the standardized name "N_DOS" shall perform the device origin setup procedure. This Program Invocation shall be permanently associated with the device origin setup domain object with the same name, as specified in subclause 8.1.

Object: Program Invocation

Key Attribute: Program Invocation Name = "N_DOS"

Attribute:	State	
Attribute:	List of Domain References	= {N_DOS, ...}
Attribute:	MMS Deletable	= FALSE
Attribute:	Reusable	= TRUE
Attribute:	Monitor	= FALSE
Attribute:	Event Condition Reference	
Attribute:	Event Action Reference	
Attribute:	Event Enrollment Reference	
Attribute:	Execution Argument	

8.3 Named Variable Objects

The name of each of the standardized NC specific Named Variable objects defined in the following subclauses, shall either be given the exact name prescribed or a name constructed in the way specified, as the case may be. The values of some of the "type description" attributes may be of an arbitrary range, in which case the range of values (as well as the size of the attribute field) shall be determined and specified by the implementer.

The generic Named Variable objects "M_powerProblem", "M_ELT", and "M_DAYTIME", defined in part 1 of ISO/IEC 9506, shall be supported if the associated function is supported.

8.3.1 NC VMD Specific Named Variable Objects

8.3.1.1 N_SWC_ – Switch Commands

The name of any "Switch_Command" variable (see clause 6.4.5) shall begin with the six character prefix "N_SWC_", complemented by a user or implementer defined identifier.

8.3.1.2 N_SWC_Remote – Command

The Named Variable object with the standardized name "N_SWC_Remote" is defined for remote switching of the NC system from the REMOTE state to LOCAL. When its value is set FALSE, then the Local/Remote State shall be switched to LOCAL CONTROL.

However, the NC system may not be remotely switched back to REMOTE CONTROL.

Object: Named Variable

Key Attribute: Variable Name = VMD-specific "N_SWC_Remote"

Attribute: MMS Deletable = FALSE
Attribute: Type Description = boolean
Attribute: Access Method = locally defined

8.3.1.3 N_SWP_ – Switch Position

The name of any "Switch_Position" variable (see clause 6.4.5) shall begin with the six character prefix "N_SWP_", complemented by a user or implementer defined identifier.

8.3.1.4 N_SWP_Remote – State Query

The Named Variable object with the standardized name "N_SWP_Remote" is defined for the NC-CS user to query the NC about the Local/Remote State. Its values correspond to those of N_SWC_Remote.

Object: Named Variable

Key Attribute: Variable Name = VMD-specific "N_SWP_Remote"

Attribute: MMS Deletable = FALSE
Attribute: Type Description = boolean
Attribute: Access Method = locally defined

The NC-CS user shall have only read access to this variable.

8.3.1.5 N_INF_ – Basic Information

The names of the Named Variable objects designed to provide basic information about the NC system shall begin with the six character prefix "N_INF_", complemented by a user or implementer defined identifier.

Object: Named Variable

Key Attribute: Variable Name = VMD-specific "N_INF_...".

Attribute: MMS Deletable = FALSE
Attribute: Type Description = as required by the implementation
Attribute: Access Method = locally defined

8.3.2 NC Domain Specific Named Variable Objects

The following Named Variable objects shall be of domain-specific scope. More specifically, the scope is that of the device domain, because these objects are device related. Because of this, the name of the corresponding device domain shall be included in the object name.

8.3.2.1 N_NumAxes - Number Of Machine Axes

The Named Variable object with the standardized name "N_NumAxes", which contains the "number of axes" value, shall be a static object of the device with which it is associated. It may be used within other configuration related data structures of the device.

Object: Named Variable

```
Key Attribute: Variable Name            = domain-specific {
                                              domainID      "N_DEV_...",
                                              itemID        "N_NumAxes"}
      Attribute:   MMS deletable           = FALSE
      Attribute:   Type Description        = unsigned 8
      Attribute:   Access Method           = locally defined
```

The NC-CS user shall have only read access to this variable.

8.3.2.2 N_ZeroAxes - Axes Zero Offsets

The Named Variable object with the standardized name "N_ZeroAxes" contains the zero offset values for all axes of the device. The number of zero offsets supported within an NC system is implementation dependent, and (along with the number of axes) it shall be specified for each device in the appropriate PICS entry.

Object: Named Variable

```
Key Attribute: Variable Name            = domain-specific {
                                              domainID      "N_DEV_...",
                                              itemID        "N_ZeroAxes"}
      Attribute:   MMS Deletable           = FALSE
      Attribute:   Type Description        = structure {
            components {
               {componentName  "on",        -- Zero offsets on
                componentType  boolean},    -- Switch for all axes
               {componentName   "zeroOffsets",
                  componentType    array {
                    numberOfElements = n-NumAxes-value
                                            -- one offset per axis
                    elementType   structure {
                       components {
                          {componentName  "nameOfAxis",
                           componentType  visible-string 1 },
                                            -- "x", "y", "z", etc.
                          {componentName  "on",
                                            -- Zero offsets on for
                                            -- this axis
                           componentType  boolean},
                          {componentName  "zeroOffsets",
                               componentType  array {
                                         numberOfElements = number-of-zero-offsets,
                                                -- device specific constant
                                         elementType  structure {
                                            components {
                                            {componentName  "on",
                                                 -- zero offset on/off
                                             componentType    boolean},
                                            {componentName  "zeroOffset",
                                                 -- zero offset value
                                             componentType  floating-point {
                                                         format-width 32,
                                                         exponent-width 8}
                  }}}}}}}}}}}
   Attribute:      Access Method            = locally defined
```

8.3.2.3 N_AxisValTrans - Axis Transformation

The Named Variable object with the standardized name "N_AxisTrans" contains the axis value transformation matrix for each axis of the device. The number of axis value transformations supported within an NC system is implementation dependent, and (along with the number of axes) it shall be specified for each device in the appropriate PICS entry.

NOTE - The meaning of the components value1, and value3 depends on the current type of transformation indentified by the value of the component "primitive". Thus, for a transition the 3 components represent a vector in terms of the coordinates x, y, ans z. In case of scaling, a scale factor may be specified for each coordinate axis. In case of mirroring value1, value2, and value3 identify mirroring about the xy plane, yz plane, and zx plane respectively. Only the values 0 and 1 shall be valid in this case, such that a value of 1 means mirroring about the indicated plane.

Object: Named Variable

```
Key Attribute: Variable Name                   = domain-specific {
                                                      domainID        "N_DEV_...",
                                                      itemID          "N_AxisTrans" }
    Attribute:    MMS Deletable                    = FALSE
    Attribute:    Type Description                 = structure {
       components {
          {componentName  "on",                 -- Transformations on
           componentType   boolean },           -- Switch for all axes
          {componentName  "transformations",
             componentType   array {
                numberOfElements = n-NumAxes-value,
                                                -- one per axis
             elementType  structure {
                components {
                   {componentName   "nameOfAxis",
                    componentType    visible-string 1 },
                                                -- "x", "y", etc.
                  {componentName   "on",        -- transformation on/off for axis
                   componentType    boolean},
                  {componentName    "transformationSequence",
                   componentType    array {     -- elements in the order of
                                                -- application
                     numberOfElements = number-of-axis-value-transformations,
                                                -- implementation dependent
                     elementType  structure {
                        components {
                           {componentName  "on", -- primitive on/off
                            componentType   boolean},
                           {componentName  "primitive",
                            componentType  unsigned 8 },
                                                -- Translation(0),
                                                -- Scaling(1),
                                                -- Mirroring(2)
                          {componentName  "value",              -- parameter value
                           componentType  floating-point {      -- ignored if M()
                                            format-width 32,
                                            exponent-width 8 } } } } }
                     } } } } } } }
   Attribute:      Access Method                  = locally defined
```

8.3.2.4 N_Machine_CS – Machine Coordinate System

The Named Variable object with the standardized name "N_Machine_CS" shall be used as the origin of the machine coordinate system.

Object: Named Variable

```
Key Attribute: Variable Name              = domain-specific {
                                                 domainID        "N_DEV_...",
                                                 itemID          "N_Machine_CS" }

     Attribute:    MMS Deletable                = FALSE
     Attribute:    Type Description             = structure {
        components {
           {componentName     "x",
            componentType     floating-point {
                                format-width 32,
                                exponent-width 8} },
           {componentName     "y",
            componentType     floating-point {
                                format-width  32,
                                exponent-width 8} },
            {componentName "z",
               componentType    floating-point {
                                format-width  32,
                                exponent-width 8} }
                      } }
     Attribute:    Access Method                = locally defined
```

8.3.2.5 N_ActProg_CS – Active Program Coordinate System

The Named Variable object with the standardized name "N_ActProg_CS" contains the identifier for the coordinate system of the currently active, or to be activated, machine program.

Object: Named Variable

```
Key Attribute: Variable Name              = domain-specific {
                                                 domainID        "N_DEV_...",
                                                 itemID          "N_ ActProg_CS"}

     Attribute:    MMS Deletable                = FALSE
     Attribute:    Type Description             = visible-string -32
                   – references the actual coordinate system, which may be one of
                   – the following:
                   –    N_UNDEFINED_CS if undefined
                   –    N_DEFAULT_CS for default coordinate system
                   –    N_MACHINE_CS for machine coordinate system
                   –    Any other, such as N_Pallet1_CS for pallet 1 coordinate
                   –    system
     Attribute:    Access Method                = locally defined
```

8.3.2.6 N_GeometryTrans – Geometry Transformation

The Named Variable object with the standardized name "N_GeometryTrans" contains the geometry transformation matrix for the device. The number of geometry transformations supported within an NC system is implementation dependent.

NOTE - If an NC supports only mirroring, for example, the array value "number-of-geometriy-transformations" within the structure of the N_GeometryTrans object shall be set to 1 and the component value "primitive" to 2 (mirroring), which should be READONLY. How to enter this component value shall be a local matter. If an NC supports only a predefined sequence of primitives, a write access to change the order of the primitives shall fail.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                    domainID       "N_DEV_...",
                                                    itemID         "N_GeometryTrans" }

      Attribute:     MMS Deletable                 = FALSE
      Attribute:     Type Description              = structure {
         components {
            {componentName   "on",                 -- Transformation on
             componentType    boolean},
            {componentName   "transformations",
             componentType   array {
                 numberOfElements = number-of-geometry-transformations,
                                                   -- implementation dependent.
                 elementType  structure {          -- Elements in the exact
                                                   -- order of transformations.
                     components {
                        {componentName  "on",  -- primitive on/off
                         componentType    boolean },
                        {componentName  "primitive",
                         componentType  unsigned 8 }
                                                   -- Translation (0),
                                                   -- Scaling (1),
                                                   -- Mirroring (2),
                                                   -- Rotation (3),
                      {componentName   "value1",
                       componentType   floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                      {componentName   "value2",
                       componentType   floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                      {componentName   "value3",
                       componentType   floating-point {
                                        format-width 32,
                                        exponent-width 8 } }
                     } } } } } }
      Attribute:     Access Method                 = locally defined
```

Note - The meaning of the components value1, value2, and value3 depends on the current value of the component primitive: In case of translation the 3 components represent a vector in terms of the coordinate system x, y, and z. In case of scaling for each coordinate axis a scaling factor may be specified. In case of rotation the angles about x, y and z axis may be specified. In case of mirroring the planes about which mirroring is effective may be selected. In this case for each component only the values 0 and 1 are valid. A value of 1 means mirroring about the plane. The xy plane may be selected by the component value1, the xz plane by the component value2, and the xz plane by the component value3.

8.3.2.7 N_FRL -- Feedrate Limit

The Named Variable object with the standardized name "N_FRL" shall contain the value of the maximum feedrate permitted for the associated device.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                    domainID       "N_DEV_...",
                                                    itemID         "N_FRL"}

      Attribute:     MMS Deletable                 = FALSE
      Attribute:     Type Description              = array {
                         numberOfElements = n_NumAxes-value,
                         elementType   floating-point {
                                          format-width 32,
                                          exponent-width  8 } }
      Attribute:     Access Method                 = locally defined
```

8.3.2.8 N_FRO_ON – Feedrate Override ON/OFF

The Named Variable object with the standardized name "N_FRO_ON" shall be used to activate feedrate override. If its value is TRUE, the current value contained in the variable N_FRO shall be applied to the program feedrate. Otherwise the local override value shall be active.

Object: Named Variable

Key Attribute: Variable Name = domain-specific {
domainID "N_DEV_...",
itemID "N_FRO_ON"}

Attribute: MMS Deletable = FALSE
Attribute: Type Description = boolean
Attribute: Access Method = locally defined

8.3.2.9 N_FRO - Feedrate Override

The Named Variable object with the standardized name "N_FRO" shall contain the feedrate override, in percent or value, to be applied to the programmed feedrate.

Object: Named Variable

Key Attribute: Variable Name = domain-specific {
domainID "N_DEV_...",
itemID "N_FRO"}

Attribute: MMS Deletable = FALSE
Attribute: Type Description = structure {
components {
{componentName "overridePerCent",
componentType boolean}, – if TRUE then
– value given in percent
{componentName "overrideValue",
componentType floating-point {
format-width 32,
exponent-width 8 } } } }
Attribute: Access Method = locally defined

8.3.2.10 N_OSP – Optional Stop

The Named Variable object with the standardized name "N_OSP" shall be used to activate the optional stop mode. If its value is TRUE, the optional stop mode shall be active.

Object: Named Variable

Key Attribute: Variable Name = domain-specific {
domainID "N_DEV_...",
itemID "N_OSP"}

Attribute: MMS Deletable = FALSE
Attribute: Type Description = boolean
Attribute: Access Method = locally defined

8.3.2.11 N_RPO_ON - Rapid Override ON/OFF

The Named Variable object with the standardized name "N_RPO_ON" shall be used to activate rapid override. If its value is TRUE, the current value contained in the variable N_RPO shall be applied to the program rapid feedrate. Otherwise the local default value shall be active.

Object: Named Variable

Key Attribute: Variable Name = domain-specific {
domainID "N_DEV_...",
itemID "N_ON"}

Attribute: MMS Deletable = FALSE
Attribute: Type Description = boolean
Attribute: Access Method = locally defined

8.3.2.12 N_RPO - Rapid Override

The Named Variable object with the standardized name "N_RPO" shall contain the rapid override, in percent or value, to be applied to all programmed rapid moves, for those devices which do use separate overrides for rapid moves and machining feedrates.

Object: Named Variable

```
Key Attribute: Variable Name                     = domain-specific {
                                                       domainID        "N_DEV_...",
                                                       itemID          "N_RPO" }

        Attribute:    MMS Deletable                 = FALSE
        Attribute:    Type Description              = structure {
           components {
              {componentName   "overridePerCent",
               componentType   boolean},           -- if TRUE then
                                                    -- value given in percent.
              {componentName   "overrideValue",
               componentType   floating-point {
                                             format-width 32,
                                             exponent-width 8} } } }
        Attribute:    Access Method                 = locally defined
```

8.3.2.13 N_NumSpindle - Number Of Spindles

The Named Variable object with the standardized name "N_NumSpindles", which contains the "number of spindles" value, shall be a static object of the device with which it is associated. It may be used within other spindle related, structured data objects of the device, such as spindle speed limits.

Object: Named Variable

```
Key Attribute: Variable Name                     = domain-specific {
                                                       domainID        "N_DEV_...",
                                                       itemID          "N_NumSpindle" }

        Attribute:    MMS deletable                 = FALSE
        Attribute:    Type Description              = unsigned 8
        Attribute:    Access Method                 = locally defined
```

The NC-CS user shall have only read access to this variable.

8.3.2.14 N_SSL - Spindle Speed Limit

The Named Variable object with the standardized name "N_SSL" shall contain the value of maximum spindle speed permitted by the associated device.

Object: Named Variable

```
Key Attribute: Variable Name                     = domain-specific {
                                                       domainID        "N_DEV_...",
                                                       itemID          "N_SSL" }

        Attribute:    MMS Deletable                 = FALSE
        Attribute:    Type Description              = array {
                               numberOfElements = n-NumSpindles-value,
                               elementType   floating-point {
                                                 format-width 32,
                                                 exponent-width 8} }
        Attribute:    Access Method                 = locally defined
```

8.3.2.15 N_SSO_ON - Spindle Speed Override ON/OFF

The Named Variable object with the standardized name "N_SSO_ON" shall be used to activate spindle speed override. If its value is TRUE, the current value of the variable N_SSO shall be applied to the programmed spindle speed. Otherwise the local (default) spindle speed override value shall be active.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                   domainID      "N_DEV_...",
                                                   itemID        "N_SSO_ON" }

     Attribute:    MMS Deletable                   = FALSE
     Attribute:    Type Description                = boolean
     Attribute:    Access Method                   = locally defined
```

8.3.2.16 N_SSO - Spindle Speed Override

The Named Variable object with the standardized name "N_SSO" shall contain the override value, in percent or value, to be applied to the programmed spindle speed.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                   domainID      "N_DEV_...",
                                                   itemID        "N_SSO" }

     Attribute:    MMS Deletable                   = FALSE
     Attribute:    Type Description                = array {
                             number of elements = n-NumSpindles-value,
                             elementType  structure {
                                 components {
                                     {componentName  "overridePerCent",
                                      componentType  boolean },      -- if TRUE then
                                                                      -- value given in
                                                                      -- percent
                                     {componentName  "overrideValue",
                                     componentType    floating-point {
                                                        format-width 32,
                                                        exponent-width 8 } } } } }
     Attribute:    Access Method                   = locally defined
```

8.3.2.17 N_DEV_STATE – Device State

The Named Variable object with the standardized name "N_DEV_STATE" shall contain the device state.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                   domainID      "N_DEV_...",
                                                   itemID        "N_DEV_STATE"}

     Attribute:    MMS Deletablte                  = FALSE
     Attribute:    Type Description                = unsigned 8
     Attribute:    Access Method                   = locally defined
```

Possible values of this variable are:

1 = POWER OFF
2 = READY
3 = NOT READY

The NC-CS user shall have only read access to this variable.

8.3.2.18 N_DEVINF_ – Device Information

The names of all Named Variable objects which contain basic information about the device, shall begin with the nine character prefix "N_DEVINF_", complemented by a user or implementer defined identifier.

```
Object: Named Variable
Key Attribute: Variable Name                 = domain-specific {
                                                   domainID      "N_DEV_...",
                                                   itemID        "N_DEVINF_..."}

     Attribute:    MMS Deletable                   = FALSE
     Attribute:    Type Description                = list of device information
     Attribute:    Access Method                   = locally defined
```

8.3.2.18.1 N_DEVINF_DROP – Device Origin Setup Query

The Named Variable object with the standardized name "N_DEVINF_DROP" shall indicate the calibration state of the NC device. A value of TRUE shall indicate that the NC device is calibrated.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_DEVINF_DROP" }
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= boolean	
Attribute:	Access Method	= locally defined	

8.3.2.19 N_SWC_Power_On – Power On Switch Command

The Named Variable object with the standardized name "N_SWC_Power_On" is defined to remotely switch the machine power ON or OFF. When its value is set TRUE then the machine power shall be turned ON, otherwise it shall be turned OFF.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_SWC_Power_On"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= boolean	
Attribute:	Access Method	= locally defined	

8.3.2.20 N_SWP_Power_On – Power On Switch Position Query

The Named Variable object with the standardized name "N_SWP_Power_On" shall be used to check whether the machine power is ON or OFF. The value TRUE of this variable shall indicate that machine power is ON.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_SWP_Power_On"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= boolean	
Attribute:	Access Method	= locally defined	

8.3.2.21 N_StoreType

The Named Variable object with the standardized name "N_StoreType" shall contain a visible string describing the type of storage medium, or other pertinent information about it.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_StoreType"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= visible-string -32	
Attribute:	Access Method	= locally defined	

8.3.2.22 N_MaxStoreEntries

The Named Variable object with the standardized name "N_MaxStoreEntries" shall contain an unsigned integer specifying the maximum number of data store entry objects which may be assigned to this data store.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_MaxStoreEntries"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.23 N_MaxStoreCapacity

The Named Variable object with the standardized name "N_MaxStoreCapacity" shall contain an unsigned integer specifying the maximum octet count of data which may be loaded into this data store.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_MaxStoreCapacity"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.24 N_CurrentEntries

The Named Variable object with the standardized name "N_CurrentEntries" shall contain an unsigned integer indicating the current number of data store entries assigned to this data store.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_CurrentEntries"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.25 N_RemainingEntries

The Named Variable object with the standardized name "N_RemainingEntries" shall contain an unsigned integer indicating the number of entries which remain available.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_RemainingEntries"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.26 N_CurrentCapacityUsed

The Named Variable object with the standardized name "N_CurrentCapacityUsed" shall contain an unsigned integer indicating the octet count of the data currently loaded into this data store.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_CurrentCapacityUsed"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.27 N_RemainingCapacity

The Named Variable object with the standardized name "N_RemainingCapacity" shall contain an unsigned integer indicating the octet count which is currently unused in this data store.

Object: Named Variable

Key Attribute: Variable Name		= domain-specific {	
		domainID	"N_DEV_...",
		itemID	"N_RemainingCapacity"}
Attribute:	MMS Deletable	= FALSE	
Attribute:	Type Description	= unsigned 32	
Attribute:	Access Method	= locally defined	

8.3.2.28 Data Store Entry

This part of ISO/IEC 9506 does not standardize any NC-specific Data Store Entry objects. However, the following template should be used by an implementer to describe those entries that are required by the implementation.

Object: Named Variable

```
Key Attribute: Variable Name             = domain-specific {
                                               domainID     "N_STORE_...",
                                               itemID       "..." }       -- the name of the
                                                                          -- referenced domain.
    Attribute:    MMS Deletable              = FALSE
    Attribute:    Type Description           = typeName {
                                                    domain-specific {
                                                              domainID "N_STORE_...",
                                                                         -- same as above
                                                                         -- domainID
                                              itemID  "N_DataStoreEntry" }
    Attribute:    Access Method              = locally defined
```

The components of the data store entry shall be the same as the components of the Data Store Entry Type named type object present in the NC Data Store Domain.

8.4 Scattered Access Objects

This part of ISO/IEC 9506 does not standardize any NC-specific Scattered Access objects.

8.5 Named Variable List Objects

8.5.1 NC VMD-specific Named Variable List Objects

8.5.1.1 Basic Information List

The Named Variable List object with the standardized name "N_INF" shall identify the List of Variables which contain the names of one or more Named Variable objects representing Basic Information.

Object: Named Variable List

Key Attribute: Variable List Name = VMD-specific "N_INF"

```
    Attribute:    MMS Deletable
    Attribute:    List of Variable
```

8.5.2 NC Domain-specific Named Variable List Objects

8.5.2.1 Device Information List

The Named Variable List object with the standardized name "N_DEVINF" shall identify the list of variables which contain the names of one or more Named Variable objects representing device information.

Object: Named Variable List

```
Key Attribute: Variable Name             = domain-specific {
                                               domainID     "N_DEV_...",
                                               itemID       "N_DEVINF"}
    Attribute:    MMS Deletable
    Attribute:    List of Variable
```

8.5.2.2 Data Store Entry List

The Named Variable List object with the standardized name "N_EntryList" shall contain the names of the Named Variable objects of type Data Store Entry from the specific data store.

Object: Named Variable List

```
Key Attribute: Variable Name                  = domain-specific {
                                                  domainID     "N_STORE_...",
                                                  itemID       "N_EntryList"}

     Attribute:     MMS Deletable                 = FALSE
     Attribute:     List of Variable {
                         name = domain-specific {
                                   domainID "N_STORE_...",       – same store as above.
                                   itemID   "..." }              – name of a Data
                                                                 – Store Entry object.
     }   – repeated for each Data Store Entry object in the data store.
```

8.5.2.3 Controlling Process Information List

All Named Variable List objects containing lists of "Controlling Process Information" shall have the name "N_PRINF". Each such Variable List object shall identify a list containing the names of one or more Named Variable objects representing Controlling Process Information.

Object: Named Variable List

```
Key Attribute: Variable Name                  = domain-specific {
                                                  domainID     "N_PID_...",
                                                  itemID       "N_PRINF"}

     Attribute:     MMS Deletable
     Attribute:     List of Variable
```

8.6 Named Type Objects

8.6.1 Data Store Entry Type

The Named Type object with the standardized name "N_DataStoreEntry" shall define the information content of a data store entry.

The named attributes of this object may be used to access information about a domain which is assigned to the Data Store domain which contains this named type object.

Example: Given a domain of name "N_PRG_1" assigned to a data store of name "N_STORE", then, a reference to the N_STORE (domain-specific) variable N_PRG_1.DataSize would yield the octet count assigned to the domain "N_PRG_1".

Object: Named Type

```
Key Attribute: Type Name                      = domain-specific {
                                                  domainID     "N_STORE_...",
                                                  itemID       "N_DataStoreEntry"}

     Attribute:     MMS Deletable                 = FALSE
     Attribute:     Type Description structure {
                         components {
                            {componentName    = "DataName",
                            componentType     = visible-string -32},
                            {componentName    = "DataType",
                            componentType     = visible-string -32 },
                            {componentName    = "DataSizeValid",
                            componentType     = boolean },
                            {componentName    = "DataSize",
                            componentType     = unsigned 32 },
                            {componentName    = "CreationDateValid",
                            componentType     = boolean },
                            {componentName    = "CreationDate",
                            componentType     = generalized-time},
                            {componentName    = "LastEdit",
                            componentType     = generalized-time },
                            {componentName    = "ProtectionMode",
                            componentType     = visible-string -32 },
                            {componentName    = "DataSource" componentType = visible-string -32 } } }.
```

8.7 Semaphore Objects

8.7.1 Control Token

The Token Semaphore object shall have the standardized name "N_Control". This object shall be of VMD-specific scope.

Object: Semaphore

Key Attribute: Semaphore Name = VMD-specific "N_CONTROL"

Attribute:	MMS Deletable	= FALSE
Attribute:	Class	= TOKEN
Attribute:	Number of Tokens	= 1
Attribute:	Number of Owned Tokens	
Attribute:	List of Owners	
Attribute:	List of Requesters	
Attribute:	Event Condition References (none)	

8.8 Operator Station Objects

This part of ISO/IEC 9506 does not define any names for Operator Station objects.

8.9 Event Condition Objects

This part of ISO/IEC 9506 does not define any names for Event Condition objects.

8.10 Event Action Objects

This part of ISO/IEC 9506 does not define any names for Event Action objects.

8.11 Event Enrollment Objects

This part of ISO/IEC 9506 does not define any names for Event Enrollment objects.

8.12 Journal Objects

This part of ISO/IEC 9506 does not standardize any NC-specific journal objects.

8.13 Other NC Specific Objects

This part of ISO/IEC 9506 does not standardize any other NC specific objects.

9 Conformance

9.1 Conformance Class Description

This part of ISO/IEC 9506 defines just one conformance class called NC which has the minimal functionality to establish or de-establish an association, or to reject it in case of a protocol error. Any NC system which supports the conformance class NC shall provide the following Services:

Initiate
Abort
Reject

In addition to the definition of the conformance class NC this part of ISO/IEC 9506 defines minimum requirements for each functional group listed in Section 5.2. Any NC system supporting such a functional group shall conform to these minimum requirements.

9.1.1 Minimum Requirements For The Functional Group Identify

Any NC system supporting the Identify functions shall provide the following MMS service:

Identify

9.1.2 Minimum Requirements For The Functional Group Data Transfer

Any NC system supporting Variable Transfer shall provide the following MMS services:

Read
Write

Any NC system supporting Download of Domains shall provide the following MMS services:

InitiateDownloadSequence
DownloadSegment
TerminateDownloadSequence
RequestDomainDownload

Any NC system supporting Upload of Domains shall provide the following MMS services:

InitiateUploadSequence
UploadSegment
TerminateUploadSequence
RequestDomainUpload

Any NC system supporting additionally Upload and Download of Domains to a Third Party Fileserver shall provide the following MMS services:

LoadDomainContent
StoreDomainContent

9.1.3 Minimum Requirements For The Functional Group NC Information

Any NC system supporting NC information shall provide the following MMS services:

Read

9.1.4 Minimum Requirements For The Functional Group Process Management

Any NC system supporting Process Management shall provide the following MMS service:

Start

9.1.5 Minimum Requirements For The Functional Group Operator Interface Device

Any NC system supporting a Operator Interface Device shall provide the following MMS service:

Output

9.1.6 Minimum Requirements For The Functional Group Alarm Processing

Any NC system supporting signaling of Alarms shall provide the following MMS Service:

EventNotification

Any NC system supporting acknowledgment of alarms shall provide the following MMS Service:

AcknowledgeEventNotification

9.1.7 Minimum Requirements For The Functional Group Switching Operation

Any NC system supporting Switching Operation shall provide the following MMS services:

Write
InformationReport

9.1.8 Minimum Requirements For The Functional Group NC Data Store Management

Any NC System supporting Data Store Management shall provide the following MMS Services:

Read
DeleteDomain

9.2 Restrictions On MMS Optional Parameters

This part of ISO/IEC 9506 places no restrictions on MMS optional parameters.

9.3 Conformance To Standardized Objects

This part of ISO/IEC 9506 prescribes no conformance to any standardized object. But the vendor of an NC system shall provide a list (contained in the PICS) of the standardized objects supported by it.

9.4 Addition To MMS PICS

The following extensions to the MMS PICS are required by this part of ISO/IEC 9506:

9.4.1 PICS For Functional Classes

The vendor of an NC system shall state which functional group (outlined in section 5.2) it supports.

Functional Group	supported ?
Identify Function	
Variable Transfer	
Domain Download	
Domain Upload	
Third Party Up/Download	
NC Information	
Process Management	
Operator Interaction	
Signaling of Alarms	
Acknowledgment of Alarms |	
Switching Operation	
NC Data Store Management	

9.4.2 PICS For Standardized Objects

The vendor of an NC system shall state which standardized objects it supports.

Domain Objects
N_DEV_...
N_PRG_...
N_PID_...
N_TLD_...
N_MOP_...
N_SET_...
N_SDD_...
N_PRB_...
N_DGN_...
N_DOS
N_STORE_...

Program Invocation Objects
N_DOS

Named Variable Objects (VMD-specific)
N_SWC_...
N_SWC_Remote
N_SWP_...
N_SWP_Remote
N_INF_...

Named Variable Objects (domain-specific)
N_NumAxes
N_ZeroAxes
N_AxisValTrans
N_Machine_CS
N_ActProg_CS
N_GeometryTrans
N_FRL
N_FRO
N_FRO_ON
N_OSP
N_RPO
N_RPO_ON
N_NumSpindle
N_SSL
N_SSO
N_SSO_ON
N_DEV_STATE
N_DEVINF_...
N_DEVINF_DOS
N_SWP_Power_On
N_SWC_Power_On

Named Variable Objects (domain-specific)
N_StoreType
N_MaxStoreEntries
N_MaxStoreCapacity
N_CurrentEntries
N_RemainingEntries
N_CurrentCapacityUsed
N_RemainingCapacity

Named Variable List Objects
N_DEVINF
N_EntryList
N_PRINF
N_INF

Named Type Objects
N_DataStoreEntry

Semaphore Objects
N_CONTROL

Annex A
Dynamic Download and Coded Domains

A.1 Dynamic Download

When the size of the NC program exceeds the storage capacity of the NC controller, it is necessary that the program data be downloaded dynamically, such that the execution of the program and the program downloading may take place simultaneously.

This kind of operation is sometimes known as "Behind the Tape Reader" operation, because the NC program is obtained from an external device same as from a paper tape reader. Advanced systems may make use of such a capability by using several simultaneous dynamic program downloads for simultaneous tasks.

When a dynamic download is requested, the CreateProgramInvocation-Request shall include in its list of domain references one or more linking domains whose names shall begin with the predefined prefix "N_DDL_". The content of this domain shall be the information needed for the NC system to begin the dynamic download.

NOTE - This domain may be created during the creation of the Program Invocation by local means with information derived from the name of the linking domain.

As long as the dynamic domain is in the loading state, it shall be included in the list of domains within the program invocation. Otherwise, there is no explicit link from this domain to the program invocation, because, due to MMS restrictions, the list of program invocation references attribute of the dynamic domain remains empty.

Service Procedure:
When a dynamic download is initiated via Program Invocation request, the list of domain references shall include the name of the linking domain beginning with the prefix "N_DDL_". When issuing a Start on the Program Invocation, a RequestDomainDownload for all dynamic domains shall be issued by the Program Invocation. After all InitiateDomainDownload-Requests are confirmed positive (the dynamic domains are in the loading state), a Start-Response shall be sent.

If a download is not possible, a Service-Error shall be issued with an error class VMD-state and an error code 'domain-transfer-problem'.

After receiving a DownloadSegment-Confirmation with MoreFollows = FALSE, a TerminateDownloadSequence-Request shall be issued with the 'discard' parameter containing an error class 'resource' and an error code 'memory-unavailable'. This will lead to the deletion of the dynamic domain.

The linking domain shall contain all information for the dynamic download. It may include such items as the following Named Variables with predefined names:

Object: Named Variable
Key Attribute: Variable Name = domain-specific {
domainID "N_DDL_...",
itemID "N_DomainName" }

Attribute: MMS Deletable =
Attribute: Type Description = visible-string -32
Attribute: Access Method = locally defined

Object: Named Variable
Key Attribute: Variable Name = domain-specific {
domainID "N_DDL_...",
itemID "N_FileName" }

Attribute: MMS Deletable =
Attribute: Type Description = visible-string -32
Attribute: Access Method = locally defined

Object: Named Variable
Key Attribute: Variable Name = domain-specific {
domainID "N_DDL_...",
itemID "N_TPYName" }

Attribute: MMS Deletable =
Attribute: Type Description = octet-string
Attribute: Access Method = locally defined

Any other information required for controlling the dynamic download shall be included in the same manner with user defined objects in this domain.

A.2 Coded Domains

No NC-specific parameter is defined by this standard for the MMS Domain Services. Instead, this standard defines Abstract Syntaxes for transferring coded data and assigns Object Identifiers to the Abstract Syntaxes. When using such an Abstract Syntax, the "loadData" CHOICE 'coded' shall be used. In connection with this refer to the ASN.1 standard for an explanation of the use of the EXTERNAL type.

A.2.1 Abstract Syntax for Structured Domains

If the coded choice of loadData is supported for transfer of domains containing MMS objects, this Abstract Syntax shall be used. ISO-9506-NCCS-2 (iso standard 9506 part(4) structured-domain-abstract-syntax-version1(2))

```
DEFINITIONS ::= BEGIN

Imports
   ObjectName, TypeSpecification, Data, Identifier, VariableSpecification, AlternateAccess. EC-Class, Priority, normalPrio-
   rity, Severity, normalSeverity, Unsigned32, Unsigned16, Modifier, ConfirmedServiceRequest, Transitions, AlarmAckRu-
   le, ApplicationReference
FROM MMS-General-Module-1
(iso standard 9506 part(2) mms-general-module-version1(2));


     CodedDomainContent ::= SEQUENCE OF CHOICE {
          namedVariable        [0] SEQUENCE {
                    variableName               [0]  ObjectName,
                    variableType               [1]  TypeSpecification,
                    variableValue              [2]  Data OPTIONAL
                    },
          scatteredAccess      [1] SEQUENCE {
                    componentName              [0]  Identifier OPTIONAL,
                    variableSpecification      [1]  VariableSpecification,
                    alternateAccess            [2]  AlternateAccess OPTIONAL
                    },
          namedVariableList  [2] SEQUENCE {
                    variableListName           [0]  ObjectName,
                    listOfVariable             [1]  SEQUENCE OF SEQUENCE {
                         variableSpecification           [0]  VariableSpecification,
                         alternateAccess                 [1]  AlternateAccess OPTIONAL },
                    },
          namedType            [3] SEQUENCE {
                    typeName                   [0]  ObjectName,
                    typeSpecification          [1]  TypeSpecification,
                    },
          semaphore            [4]  ObjectName,
          eventCondition       [5] SEQUENCE {
                    eventConditionName         [0]  ObjectName,
                    class                      [1]  EC-Class,
                    priority                   [2]  Priority DEFAULT normalPriority,
                    severity                   [3]  Severity DEFAULT normalSeverity,
                    alarmSummaryReports        [4]  BOOLEAN OPTIONAL,
                    monitoredVariable          [5]  VariableSpecification OPTIONAL,
                    evaluationInterval         [6]  Unsigned32 OPTIONAL
                    },
          eventAction          [6] SEQUENCE {
                    eventActionName            [0]  ObjectName,
                    listOfModifier             [1]  SEQUENCE OF Modifier OPTIONAL,
                    confirmedServiceRequest    [2]  ConfirmedServiceRequest
                    },
        eventEnrollment       [7] SEQUENCE {
                eventEnrollmentName            [0]  ObjectName,
                eventConditionName             [1]  ObjectName,
                eventConditionsTransitions     [2]  Transitions,
                alarmAcknowledgementRule       [3]  AlarmAckRule,
                eventActionName                [4]  ObjectName OPTIONAL,
                clientApplication              [5]  ApplicationReference OPTIONAL,
                acknowledgementEventCondition [6] ObjectName OPTIONAL
                },
        other data            [8] IMPLICIT OCTET STRING
        }
END
```

A.2.2 Abstract Syntax for Character Based User Programs

If the coded choice of loadData is supported for transfer of domains containing character based user programs, such as ISO 6983 programs, this Abstract Syntax shall be used. ISO-9506-NCCS-3 (iso standard 9506 part(4) nc-program-abstract-syntax-version1(3)) DEFINITIONS ::= BEGIN

CodedDomainContent ::= SEQUENCE OF VisibleString -- each VisibleString contains one line of the user program.

END

Annex B
Extension of ISO 9506-4 To Include NC Milling Machine Functionality

B.0 Introduction

This normative annex extends the general applicability of ISO 9506-4. The structure of this annex is the same as that of the main part of this standard, such that the clauses are in alignment. However, not every clause in the main part of ISO 9506-4 may have counterpart in this annex, because its scope does not require it.

B.1 Scope And Field Of Application

This Annex provides the additional objects and standardized names which may be required when this standard is used in NC milling machine applications, specifically in stand-alone usage (as opposed to integrated into a flexible manufacturing system).

B.2 References

This annex does not use any references besides those which are listed in clause 2 of ISO 9506-4.

B.3 Definitions

This annex does not use any terms besides those which are defined in clause 3 of ISO 9506-4.

B.4 Symbols and Abbreviations

This annex does not use any symbols and abbreviations besides those which are listed in clause 4 of ISO 9506-4.

B.5 Application Description

In terms of this annex, NC milling machines may consist of the axes, tools, spindles, tool magazines, and user programs which control these entities. This is illustrated in figure 26.

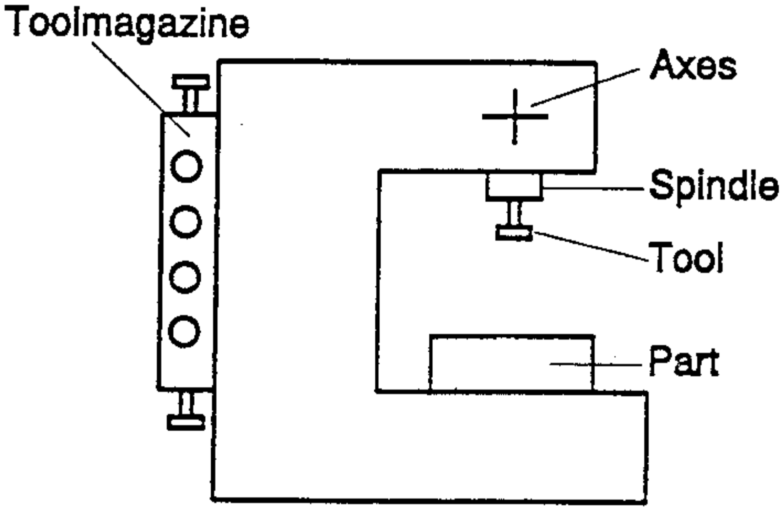


Figure 26 - NC Milling Machine

B.5.1 Tool Magazines, Tool Holders, Tools, Edges

Milling tools consist of one or more edges. For most applications, in addition to the part program data, the NC controller may also require the geometrical description of the tools used - more precisely that of their cutting edges. The objects defined in this annex take into account the geometrical relationships which exist between tool holders, tools and edges. Also defined herein are objects which model the tool magazines or pallets containing the tools.

Tool magazine, tool holders, tools, and edges are illustrated in figure 27.

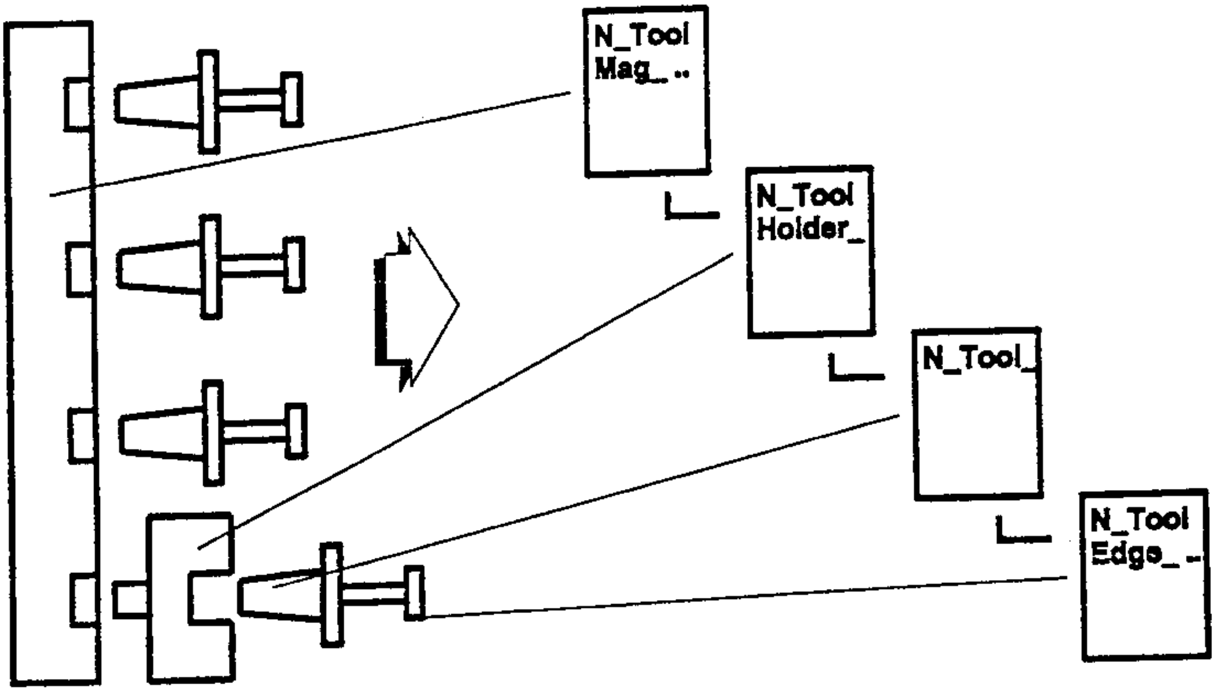


Figure 27 - Example of Tool Magazine, Tool Holder, Tools, Edges

B.6 NC Milling Specific Context Mapping

B.6.2 Objects Which Map To Domains

B.6.2.1 Tool Data Object

The Tool Data domain described in section 8.1.4 in the main part of this standard may be used for up and downloading of named variable objects describing the tool magazine, tool holder, tool, and tool edge, which are defined in this section. These objects shall be subordinated objects of the tool data domain. In accord with the definitions of the main part of this standard, more than one real NC controlled device may exist within the NC VMD. Each of these devices may have its own tool storage, or they may have a common tool storage, or both.

NOTE - In case a pallet is used for tool storage, then the means of production data domain described in clause 8.1.5 of the main part of this standard should be used for up and downloading of the named variable objects tool holder, tool, and tool edge, which in this case shall be subordinated objects of the means of production data domain.

The names of Tool Data domains shall begin with the standardized prefix N_TLD_. An example of the tool data domain is shown in figure 28.

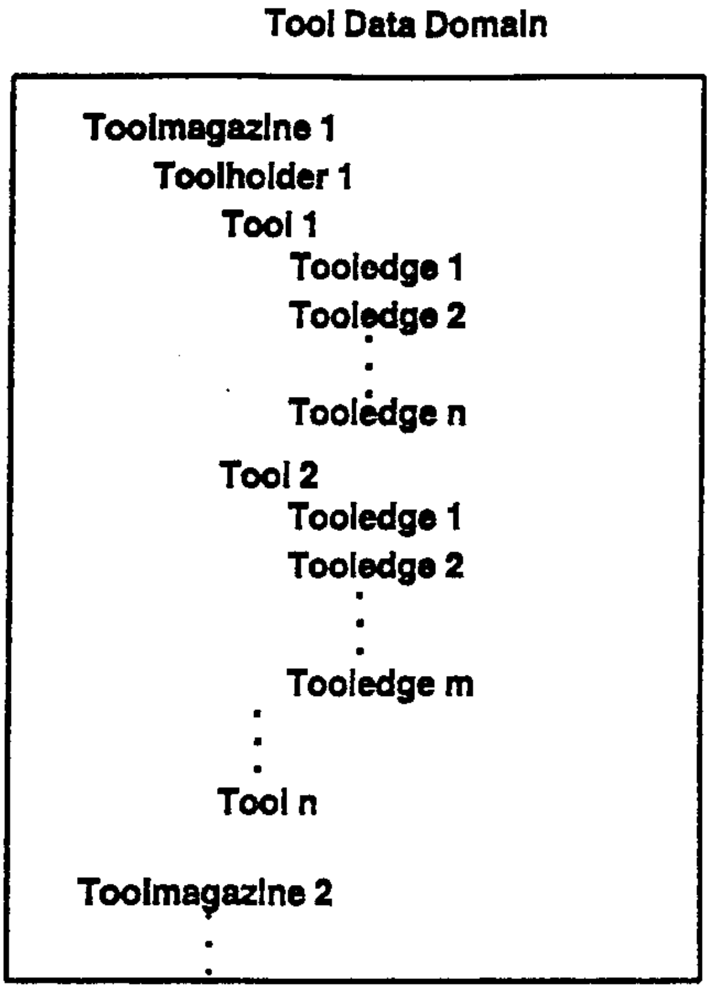


Figure 28 - Tool Data Object

B.6.4 Objects which Map to Named Variables

B.6.4.1 Tool Magazine Object

This object describes the capability to store tools or tool holders on the machine, and which includes the mechanism for automatic interchange of tools or tool holders between tool storage and machine spindle.

Key Attribute: Tool Magazine ID

Attribute:	List of Stored Objects
Attribute:	Magazine Type
Attribute:	Number of Positions

- Tool Magazine ID -- a selection from an implementation defined list of possible tool storage devices.
- List of Stored Objects -- a list of actual objects in the magazine. It shall be an array, with the array size equal to the Number of Positions.
- Magazine Type -- a selection from user defined tool magazine types.
- Number of Positions -- the total number of storage locations for tool objects or tool holder objects.

The Tool Magazine object is properly mapped into an MMS Named Variable object. The name and structure of this variable is described in clause B.8.3.2.1.

B.6.4.2 Tool Holder Object

This object contains information about a tool holder.

Key Attribute: Tool Holder ID

Attribute:	Tool Holder Type
Attribute:	Tool Holder Location ID
Attribute:	Tool Holder Position
Attribute:	List of Tools
Attribute:	Tool Holder State
Attribute:	List of Tool Holder Dimensions
Attribute:	Process Information
Attribute:	User Specific Information

- Tool Holder ID – the serial number or user code associated with this tool holder.
- Tool Holder Type -- a selection from a user defined list of tool holder types.
- Tool Holder Location ID – a back reference to the object which contains this tool holder, namely a tool magazine or a pallet.
- Tool Holder Position -- a back reference to the position in the location object, such as a tool magazine pocket number or partition of a pallet.
- List of Tools – a list of actual tool object identifiers.
- Tool Holder State – one from a list of user defined conditions.
- List of Tool Holder Dimensions – a user defined list of of tool holder measurements.
- Process Information – a user defined list of information items needed for the process.
- User Specific Information – a user defined list of information items about the tool holder.

The Tool Holder object shall be properly mapped into an MMS Named Variable object.
The name and structure of this variable is described in clause B.8.3.2.2.

B.6.4.3 Tool Object

This object contains information about a tool.

Key Attribute: Tool ID

Attribute:	Tool Type
Attribute:	Tool Location ID
Attribute:	Tool Position
Attribute:	List of Tool Edges
Attribute:	Tool State
Attribute:	List of Tool Dimensions
Attribute:	Process Information
Attribute:	Tool Handling Information

- Tool ID – serial number or user code associated with this tool.
- Tool Type – a selection from a user defined list of tool types.
- Tool Location ID -- a back reference to the object which contains this tool, namely a tool magazine, a tool holder, or a pallet.
- Tool Position – a back reference to the position within the location object, that is, tool magazine pocket number, or pallet partition.
- List of Tool Edges – an array of the edges which are a part of this tool.
- Tool State -- one of the following conditions:

  (0) In Use – the tool is operating.

  (1) Worn Out – the remaining life is zero.

  (2) Broken -- the tool has at least one broken edge.

  (3) En Route – the tool has been removed from or is en route to the magazine.

(4) Not Released – the tool is not released for part operations

(5) Released – the tool is released for part operations

(6) Needs Measurement – the tool needs measurement

(7) Needs Measurement After First Operation – the tool needs measurement after first operation

NOTE - The user may define additional states.

- List of Tool Dimensions – a user defined list of the tool measurements.
- Process Information – a user defined list of information items required for the process.
- Tool Handling Information – a user defined list of information items required for the tool handling operations.

The Tool object shall be properly mapped into an MMS Named Variable object. Name and structure of this variable is defined in clause B.8.3.2.3.

B.6.4.4 Tool Edge Object

This object defines a cutting edge. For multiple edge tools there shall be a separate such object for each edge.

Key Attribute: Edge ID

Attribute:	Tool ID
Attribute:	List of Edge Offsets
Attribute:	Nominal Edge Life
Attribute:	Remaining Edge Life
Attribute:	User Specific Information

- Edge ID – the serial number or user code associated with this edge.
- Tool ID – back reference to the Tool object which contains this edge object.
- List of Edge Offset – a user defined list of offsets.
- Nominal Edge Life – a measure of the expected life for a new or sharpened edge. This shall be the starting value for actual life.
- Remaining Edge Life – a measure of the remaining life of the edge.
- User Specific Information – a list of user defined items containing information about this edge.

The Tool Edge object shall be properly mapped into an MMS Named Variable object. Name and structure of this variable is described in clause B.8.3.2.4.

B.8 NC Milling Specific Standardized Objects

B.8.2 Standardized Domains

B.8.2.1 Tool Data Domain

The Tool Data Domain is defined in clause 8.1.4 of the main part of this standard.
Its name prefix shall be used for any domain associated with tool magazines, tool holders, tools, and tool edges.

B.8.3 Named Variable Objects

B.8.3.1 VMD-specific Standardized Named Variable Objects

B.8.3.2 Domain-specific Standardized Named Variable Objects

B.8.3.2.1 Tool Magazine Object

The name of the Tool Magazine object shall begin with the standardized prefix "N_ToolMag_", complemented by a user or implementer defined tool magazine identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name =  domain-specific {
                                      domain ID          "N_TLD_....",
                                      item ID            "N_ToolMag_..." }
      Attribute:     MMS Deletable                  =  FALSE
      Attribute:     Type Description               =  structure {
         components {
           {component Name = "StoredObjects",
            component Type = array {
                          numberOfElements,      – number of pockets
                          elementType = visible-string -32 } },
           {component Name = "MagazineType",
            component Type = visible-string -32 },
           {component Name = "NumberOfPositions",
            component Type = unsigned 16 }
         }
      }
      Attribute:     Access Method                  =  locally defined
```

B.8.3.2.2 Tool Holder Object

The name of the Tool Holder object shall begin with the standardized prefix "N_ToolHolder_", complemented by a user or implementer defined tool holder identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name =  domain-specific {
                                     domain ID          "N_TLD_...",
                                            or          "N_MOP_...",
                                     item ID            "N_ToolHolder_..." }
      Attribute: MMS Deletable   = FALSE
      Attribute: Type Description  = structure {
         components {
           {component Name = "HolderType",
            component Type = visible-string -32 },
           {component Name = "Location",
            component Type = visible-string -32 },
           {component Name = "Position",
            component Type = visible-string -32 },
           {component Name = "HolderTools",
            component Type = array {
                   numberOfElements,      – number of location
                   elementType = visible-string -32 } },
           {component Name = "HolderState",
            component Type = unsigned 8   – value from a list                  of user defined states
                                                                                },
           {component Name = "HolderDimensions",
            component Type = array {
                   numberOfElements,      – number of holder                    dimensions
                   elementType = floating-point {
                                format-width 32,
                                exponent-width 8 } },
                 {component Name = "ProcessInformation",
                 component Type      – user defined },
                 {component Name = "UserInformation",
                 component Type      – user defined }
             }
           }
   Attribute:   Access Method = locally defined
```

B.8.3.2.3 Tool Object

The name of the Tool object shall begin with the standardized prefix "N_Tool_", complemented by a user or implementer defined tool identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name = domain-specific {
                                    domain ID        "N_TLD_...",
                                           or        "N_MOP_...",
                                    item ID          "N_Tool_..." }
    Attribute:    MMS Deletable                  = FALSE
    Attribute:    Type Description               = structure {
      components {
        {component Name = "ToolType",
         component Type = visible-string },
        {component Name = "Location",
         component Type = visible-string -32 },
        {component Name = "Position",
         component Type = visible-string },
        {component Name = "ToolEdges",
         component Type = array {
              numberOfElements,     -- number of tool edge locations
              elementType = visible-string -32 } },
        {component Name = "ToolState",
         component Type = unsigned 8   -- 0 In Use
                                       -- 1 Worn
                                       -- 2 Broken
                                       -- 3 En Route
                                       -- 4 Not Released
                                       -- 5 Released
                                       -- 6 Needs Measurement
                                       -- 7 Needs Measurement After
                                       -- :  First Operation
                                       -- n User defined state(s)
                                                 },
            {component Name = "ToolDimensions",
            component Type = array {
              numberOfElements,     -- number of tool dimensions
              elementType = floating-point {
                format-width 32,
                exponent-width 8 } } },
            {component Name = "ProcessInformation",
            component Type     -- user defined },
            {component Name = "HandlingInformation",
            component Type     -- user defined }
        }
      }
   Attribute:     Access Method    = locally defined
```

B.8.3.2.4 Tool Edge Object

The name of the Tool Edge object shall begin with the standardized prefix "N_ToolEdge_", complemented by a user or implementer defined tool edge identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name = domain-specific {
                                    domain ID      "N_TLD_...",
                                           or      "N_MOP_...",
                                    item ID        "N_ToolEdge_..." }
      Attribute:    MMS Deletable                  = FALSE
      Attribute:    Type Description               = structure {
         components {
            {component Name = "ToolID",
             component Type = visible-string -32 },
            {component Name = "EdgeOffsets",
             component Type = array {
                   numberOfElements,       -- number of edge offsets
                   elementType = floating-point {
                     format-width 32,
                     exponent-width 8 } } },
            {component Name = "NominalEdgeLife",
             component Type = unsigned 32 },
            {component Name = "RemainingEdgeLife",
             component Type = unsigned 32 },
            {component Name = "UserInformation",
             component Type    -- user defined }
            }
      }
      Attribute:    Access Method                  = locally defined
```

B.9 Conformance

B.9.4 PICS for standardized objects

Domain specific Named Variable Objects
N_ToolMag_...
N_ToolHolder_...
N_Tool_...
N_ToolEdge_...

Annex C
Extension Of ISO 9506-4 To Include Turning Machine Functionality

C.0 Introduction

This normative annex extends the general applicability of ISO 9506-4.
The structure of this annex is the same as that of the main part of this standard, such that the clauses are in alignment. However, not every clause in the main part of ISO 9506-4 may have counterpart in this annex, because its scope does not require it.

C.1 Scope and field of application

This annex provides the additional objects and standardized names which may be required when this standard is used in NC turning machine applications, specifically in stand-alone usage (as opposed to integrated into a flexible manufacturing system).

C.2 References

This annex does not use any references besides those which are listed in clause 2 of ISO 9506-4.

C.3 Definitions

This annex does not use any terms besides those which are defined in clause 3 of ISO 9506-4.

C.4 Symbols and Abbreviations

This annex does not use any symbols and abbreviations besides those which are listed in clause 4 of ISO 9506-4.

C.5 Application Description

In terms of this annex, NC turning machines may consist of the machine axes, tools, spindles, tool magazines, and user programs which control these entities. This is illustrated in figure 29.

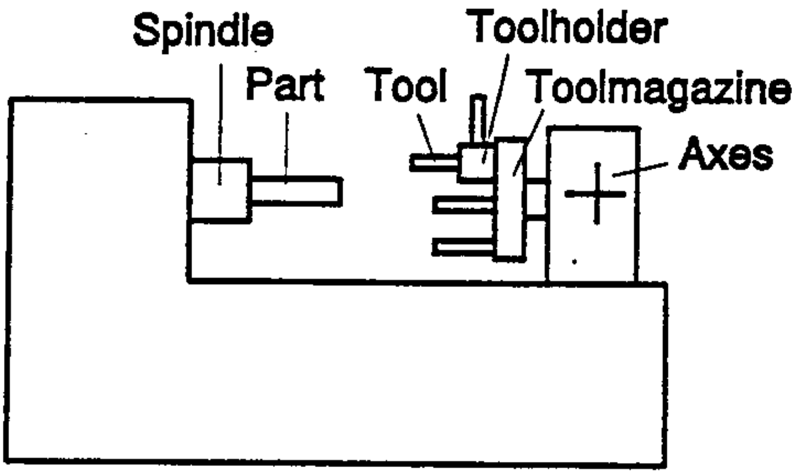


Figure 29 - Turning Machine

The objects Tool Magazine, Tool Holder, Tools, and Tool Edges for the turning machine are in concept identical to those used on milling machines. Therefore, the definitions of these objects in Annex B also apply to this Annex C.

C.5.1 Part Setup

In terms of the NC turning machine, part setup refers the combination of chuck jaws, part fixturing, and the mechanism used to position the part in the machine's work area. Portions of the setup may be re-configured for different parts or groups of parts, and may be changed manually or automatically.

C.5.2 Taper Trims

The taper trims are offsets which are provided to be applied when long flexible parts are being machined to prevent taper cuts induced when the unsupported section of a part is deflected by the tool.

C.6 NC Turning Specific Context Mapping

C.6.4.1 Objects Which Map To Named Variables

C.6.4.1.1 Part Setup Object

The purpose of the setup object is to define the configuration of chuck jaws, fixture, and part which may occupy the work area of the machine.

Key Attribute: Setup ID

Attribute:	Chuck ID
Attribute:	Chuck Size
Attribute:	Setup Objects
Attribute:	Setup State
Attribute:	Setup Offsets

- Setup ID -- serial number or alphanumeric string which identifies the setup.
- Chuck ID -- serial number or alphanumeric string which identifies the chuck used with this setup.
- Chuck Size -- provides the maximum dimensions of the chuck.
- Setup Objects -- provides a list of fixture or part objects which are or may be used in this setup.
- Setup State -- one of the following:

1. Empty	- There are no parts in this setup
2. Ready	- This setup is ready to run
3. Finished	- The parts associated with this setup are finished
4. Manual	- This setup requires manual intervention
5. Fault	- A fault has been detected with this setup

- Setup Offsets -- a measurement of the distance from the origin of the machine work area to the setup origin.

The Part Setup object is properly mapped into an MMS Named Variable object.
Name and structure of this variable is described in clause C.8.3.1.1.

C.6.4.1.2 Taper Trims Object

The pupose of the taper trims object is to provide the offsets required to compensate for deflection of cantilevered workpieces.

Key Attribute: Taper Trim ID

Attribute:	X Axis Trim
Attribute:	Z Axis Trim

- Taper Trim ID -- serial number or alphanumeric string which identifies this variable.
- X Axis Trim -- provides the offset dimension in X.
- Z Axis Trim -- provides the offset dimension in Z.

The Taper Trims object is properly mapped into an MMS Named Variable object.
Name and structure of this variable is described in clause C.8.3.1.2.

C.8.3 Named Variable Objects

C.8.3.1 VMD-specific Standardized Named Variable Objects

C.8.3.1.1 Setup Object

The name of any "setup" named variable object shall begin with the prefix "N_Setup_", complemented by a user or implementer defined identifier.

```
OBJECT: Named Variable
Key Attribute:    Variable Name = domain-specific {
                                    domain ID        "N_SET_....",
                                    item ID          "N_Setup_ ..." }
      Attribute: MMS Deletable   = FALSE
      Attribute: Type Description = structure {
         components {
           {component Name = "ChuckID",
            component Type = visible-string },
           {component Name = "ChuckSize",
            component Type = array {
                    numberOfElements, -- number of chuck dimensions
                    elementType = unsigned 16 } },
           {component Name = "SetupObjects",
            component Type = array {
                    numberOfElements, -- number of objects
                    elementType = visisble-string } },
           {component Name = "SetupState",
            component Type = unsigned 8    -- 1 Empty
                                           -- 2 Ready
                                           -- 3 Finished
                                           -- 4 Manual
                                           -- 5 Fault
                                           },
           {component Name = "SetupOffsets",
            component Type = structure {
                         components {
                              {component Name = "XComponent",
                              component Type = floating-point {
                                           format-width 32,
                                           exponent-width 8 },
                             {component Name = "ZComponent",
                             component Type = floating-point {
                                          format-width 32,
                                          exponent-width 8 } }
          } } } } } } }
      Attribute:  Access Method   = locally defined
```

C.8.3.1.2 Taper Trims Object

The name of any "taper trim" named variable object shall begin with the prefix "N_TaperTrims_", complemented by a user or implementer defined identifier.

```
OBJECT: Named Variable
Key Attribute:    Variable Name = domain-specific {
                                    domain ID        " N_SET_...",
                                    item ID          " N_TaperTrims_..." }
      Attribute: MMS Deletable   = FALSE
      Attribute: Type Description = structure {
         components {
           {component Name = "XTrim",
            component Type = floating-point {
                             format-width 32,
                             exponent-width 8 } },
          {component Name = "ZTrim",
           component Type = floating-point {
                            format-width 32,
                            exponent-width 8 } }
          } }
      Attribute:  Access Method   = locally defined
```

C.9 Conformance

C.9.4 PICS for standardized objects

Domain specific Named Variable Objects
N_ToolMag_...
N_ToolHolder_...
N_Tool_...
N_ToolEdge_...
N_Setup_...
N_TaperTrims_...

Annex D
Extension Of ISO 9506-4 To Include Flexible System Functionality

D.0 Introduction

This normative annex extends the general applicability of ISO 9506-4.
The structure of this annex is the same as that of the main part of this standard, such that the clauses are in alignment. However, not every clause in the main part of ISO 9506-4 may have counterpart in this annex, because its scope does not require it.

D.1 Scope and field of application

This annex provides the additional objects and standardized names which may be required when this standard is used in flexible systems applications, specifically when milling or turning machines, or both are integrated into a flexible manufacturing system.

D.2 References

This annex does not use any references besides those which are listed in clause 2 of ISO 9506-4.

D.3 Definitions

This annex does not use any terms besides those which are defined in clause 3 of ISO 9506-4.

D.4 Symbols and Abbreviations

This annex does not use any symbols and abbreviations besides those which are listed in clause 4 of ISO 9506-4.

D.5 Application Description

Milling or turning machines may be integrated into flexible manufacturing systems.
The most important characteristic of such a system is automated transportation of parts, tools, and other means of production, to and from machines with automatic loading and unloading devices. Pallets and fixtures may be used for this transportation of parts and tools. These objects take part in the machining process controlled by an NC controller. All these objects may be stored and manipulated on a machine, they may be located in a pallet storage, a transfer station, the working area of a machine, or similar locations. Figure 30 shows a milling machine integrated into a flexible manufacturing system.

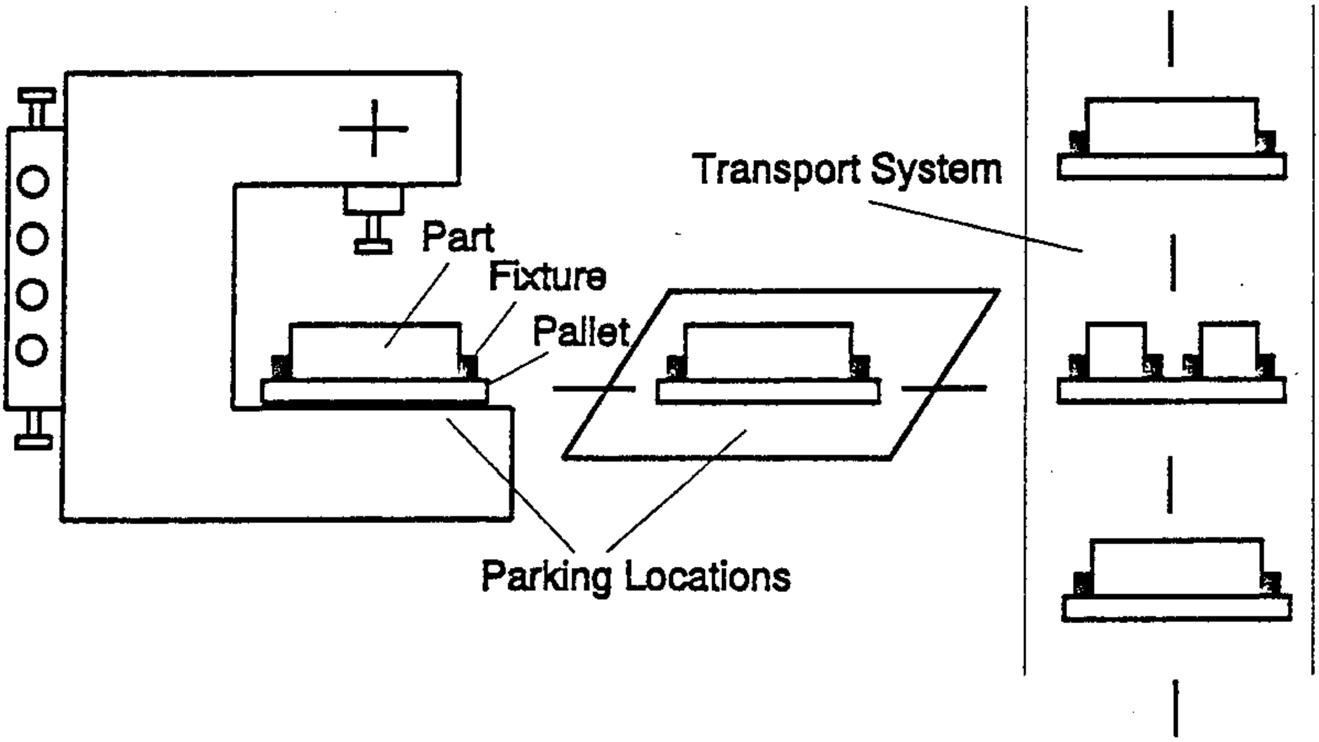


Figure 30 - Milling Machine in a Flexible Manufacturing System

During the manufacturing process a robot, an auxiliary device, or an operator may exchange new tools for used ones in the tool magazine. An automated transport system, or an operator carries new parts to the machine and removes finished parts from the machine. The data for the exchanged tools, the parts delivered to the machine, the offsets for the pallets, fixtures, parts and other informations needs to be transferred between the host system or a file store and the NC system. Therefore, it is necessary that these components which become part of the NC system during the manufacturing process, shall be mapped to MMS objects.

D.5.1 Parking locations, partitions

During an automated manufacturing process the means of production, such as pallets, parts, or similar, required objects usually are located at specific places within the work space of the system. This standard defines these places as parking locations. A parking location may be subdivided into partitions, as illustrated in figure 31. A pallet or other means of production may be located at a parking location or within a partition of a parking location, and for the automated manufacturing process it is necessary to define such location accurately.

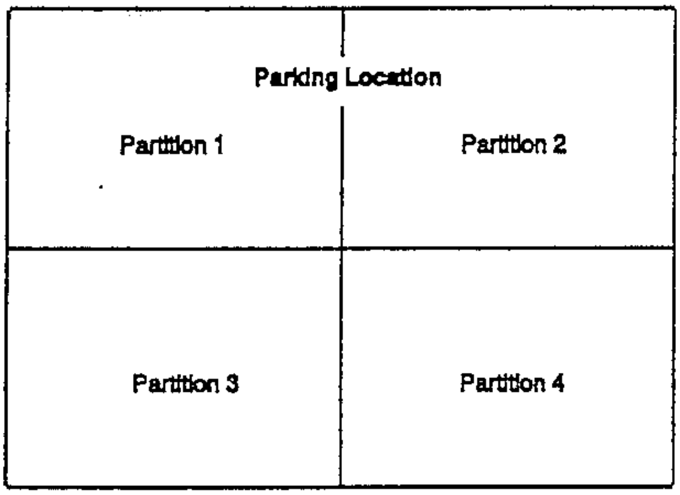


Figure 31 - Example for a Parking Location with 4 Partitions

D.5.2 Pallets, partitions, fixtures, parts

For the transport of the means of production pallets are normally used. A Pallet may also be subdivided into partitions. Figure 32 illustrates a pallet subdivided into 4 Partitions. Figure 33 illustrates a pallet located within a partition of a parking location.

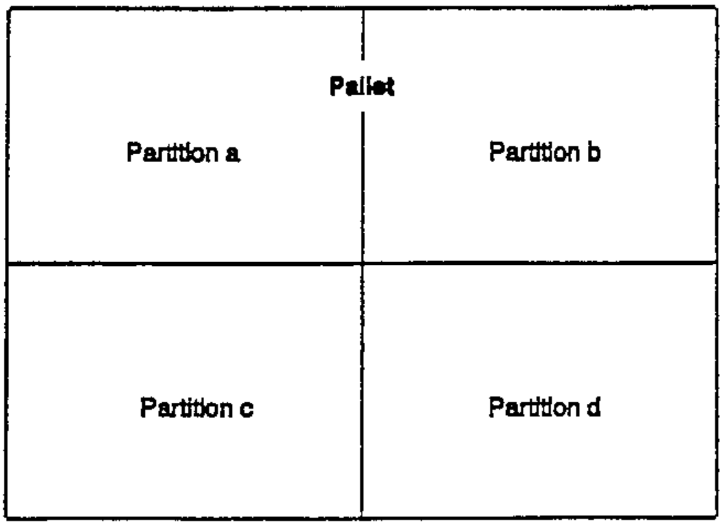


Figure 32 - Example of a Pallet with 4 Partitions

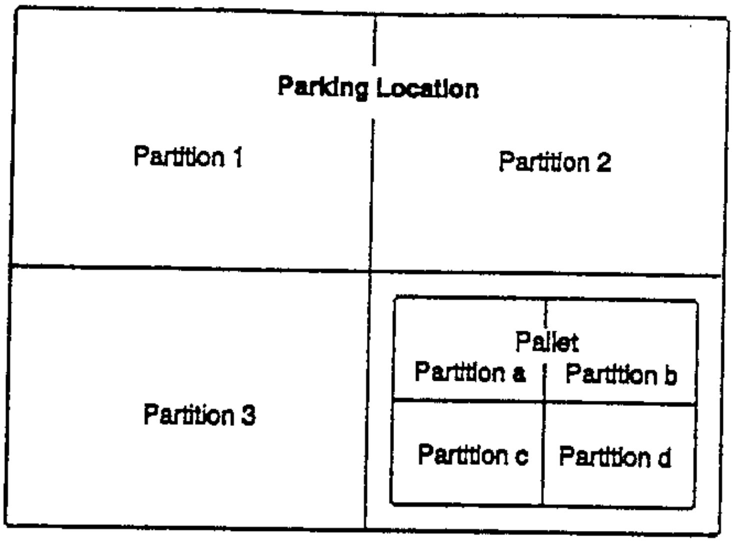


Figure 33 - Example of a Parking Location with a Pallet

In addition to parts, tools or other means of production, pallets may also contain fixtures for the parts.

For the automated manufacturing process it is necessary to determine the position of the means of production precisely, such that for example, the position of a part mounted on a pallet located within the working envelope of the machine may be determined. It is necessary that the relationship of parking location, partition, pallet, fixture and part be available to the NC controller, and this shall be considered when mapping these objects to MMS objects. An example of such a relationship is illustrated in figure 34. There may be many other such combinations of object relationships.

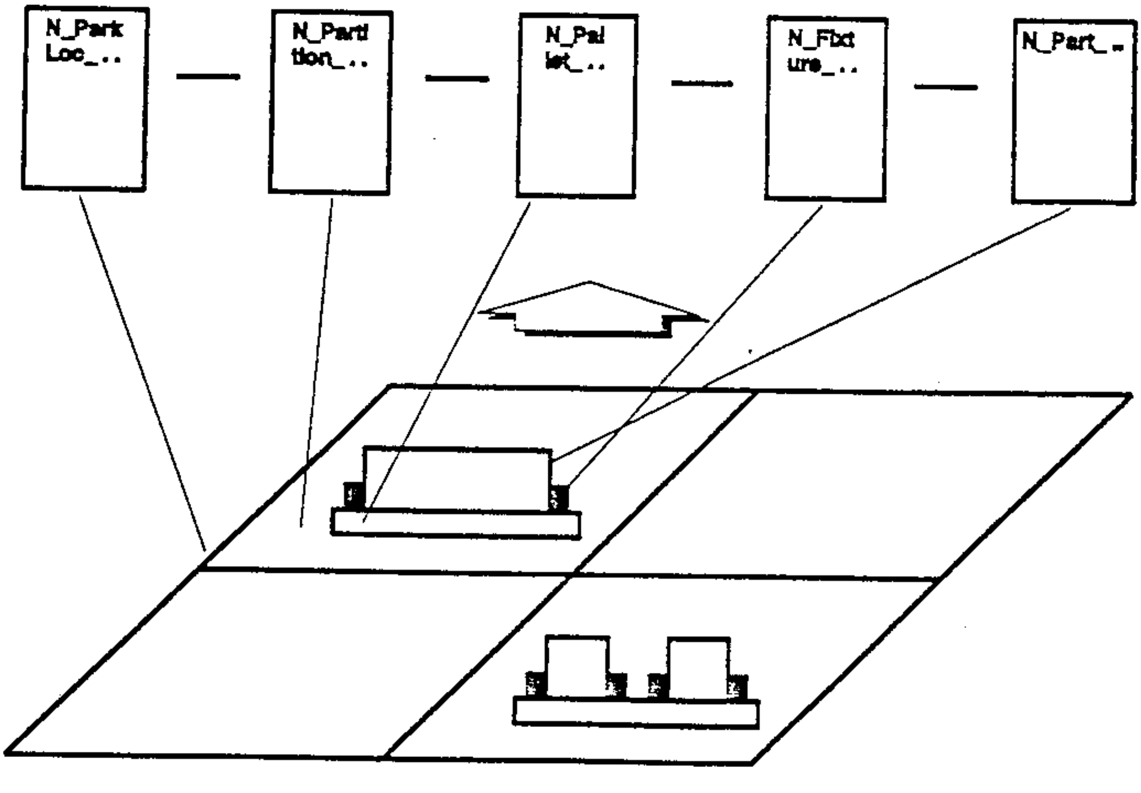


Figure 34 - Example of Parking Location, Partition, Pallet, Fixture, and Part Relationship

D.6 Flexible Systems Specific Context Mapping

D.6.2 Objects Which Map To Domains

D.6.2.1 Means of Production Data

The Means of Production Data Domain Object contains data which describe the means of production, such as parking locations, pallets and all the objects carried by such pallets, which may include fixtures, parts, and similar objects. This domain provides a naming space for Domain-specific Named Variables. An example of the Means of Production Data Domain is illustrated in figure 35.

For general definitions of these domains refer to chapter 6.2.1 in the main part of this standard.

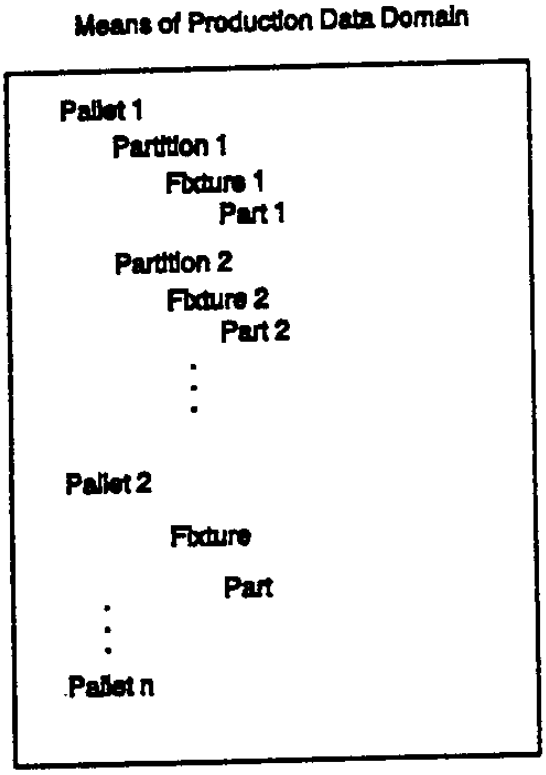


Figure 35 - Means of Production Data Object

D.6.4 Objects Which Map To Named Variables

D.6.4.1 Parking Location Object

The Parking Location object is used to define a pallet stand, machine table, storage rack, or similar holding place for pallets, fixtures, parts, tools, and similar objects.

Key attribute: Location ID

Attribute:	Location Type
Attribute:	Location State
Attribute:	List of Present Objects

- Location ID – a unique name or character string which identifies the parking location.
- Location Type – a selection from a user defined list of possible types of places which may hold objects, such as 'pallet storage of machine X'.
- Location State – one of the following:

(0)	EMPTY	– no object resides at this parking location
(1)	not FULL	– spaces for objects are available
(2)	FULL	– no space for additional objects available

- List of Present Objects – a list of objects which are present at this parking location

D.6.4.2 Partition Object

The Partition Object shall provide the means to subdivide a parking location or pallet into one or more locations which may hold physical objects.

Key attribute: Partition ID

Attribute:	Partition State
Attribute:	Object Present ID
Attribute:	Partition Location
Attribute:	Offset Reference
Attribute:	Partition Offset

- Partition ID – a unique name or character string which identifies the partition.
- Partition State – one of the following:

  (0) EMPTY – no object resides at this partition

  (1) RESERVED – this partition is reserved for a defined object

  (2) OCCUPIED – an object is present at this partition
- Object Present ID – the identifier of the object present at this partition.
- Partition Location – a back reference to the object which contains this partition.
- Offset Reference – identifies the coordinate system on which the partition offset is based, such as the coordinate system of the machine, of a parking location, or of a pallet.
- Partition Offset – defines the geometric relationship of the partition coordinate system relative to the Offset Reference coordinate system, which may consist of both translation and rotation dimensions.

NOTE - The order of the partition offset transformations shall follow the specification of clause D.8.3.2.2.

D.6.4.3 Pallet Object

The purpose of the Pallet Object is to support parts, tools, or other objects securely, for transport to and from workstations in the automated factory. This allows several stations or processes to operate on a given part without the need for refixturing or relocating the part.

Key Attribute: Pallet ID

Attribute:	Pallet Type
Attribute:	Pallet State
Attribute:	List of Objects Present
Attribute:	Pallet Location
Attribute:	Offset Reference
Attribute:	List of Pallet Offsets

- Pallet ID – the serial number or alphanumeric string associated with each pallet.
- Pallet Type -- a selection from user defined pallet types.
- Pallet State – one of the following:

  (0) EMPTY - no object resides on this pallet

  (1) not FULL - space available for additional objects

  (2) FULL - no space available for additional objects

NOTE - The user may define additional states.

- List of Objects Present – provides a list of objects which are present on the pallet.
- Pallet Location -- a back reference to the object which contains this pallet, such as a parking location.
- Offset Reference – identifies the coordinate system on which the pallet offset is based, such as the coordinate system of the machine or of a parking location.
- Pallet Offset -- defines the geometric relationship of the pallet coordinate system relative to the Offset Reference coordinate system, which may consist of both translation and rotation dimensions.

NOTE - The order of the pallet offset transformations shall follow the specification of clause D.8.3.2.3.

D.6.4.4 Fixture Object

Fixture objects define the exact location of parts, tools, or other means of production mounted on a pallet.

Key Attribute: Fixture ID

Attribute: Fixture Type
Attribute: Fixture State
Attribute: Object Present ID
Attribute: Fixture Location
Attribute: Offset Reference
Attribute: List of Fixture Offsets

- Fixture ID -- the identifier associated with each fixture.
- Fixture Type – a selection from a user defined list of fixture types.
- Fixture State -- one of a list of user defined conditions.
- Object Present ID -- a reference to the fixtured object, for example a part.
- Fixture Location – a back reference to the object which contains this fixture.
- Offset Reference – identifies the coordinate system on which the fixture offset is based, such as the coordinate system of the machine, of a parking location, or of a pallet.
- Fixture Offset – defines the geometric relationship of the fixture coordinate system relative to the Offset Reference coordinate system, which may consist of both translation and rotation dimensions.

NOTE - The order of the fixture offset transformations shall follow the specification of clause D.8.3.2.4.

D.6.4.5 Part Object

The part object is the atomic unit of work which is the whole objective of the machining process. This object should provide a means of tracking that unit of work through the manufacturing process, and provide for monitoring its status. The object should fully define the required operations necessary to complete the part to the required specification.

Key Attribute: Part ID

Attribute: Part Type
Attribute: Part State
Attribute: List of Part Data
Attribute: Part Location
Attribute: Offset Reference
Attribute: List of Part Offsets

- Part ID -- the identifier associated with each part.
- Part Type -- a selection from a user defined list of part types.
- Part State -- one of the following:

(0) Raw – original state
(1) Modified – some work done
(2) Finished – completed with no errors
(3) Scrapped – aborted with incorrectable error.
(4) Rework – finished with correctable error.

NOTE - The user may define additional states.

- List of Part Data -- a user defined list of data for part operations.
- Part Location -- a back reference to the object which contains this part.
- Offset Reference -- identifies the coordinate system on which the part offset is based, such as the coordinate system of the machine, of a parking location, or of a pallet.
- Part Offset -- defines the geometric relationship of the part coordinate system relative to the Offset Reference coordinate system, which may consist of both translation and rotation dimensions.

NOTE - The order of the part offset transformations shall follow the specification of clause D.8.3.2.5.

D.8 Flexible Systems Specific Standardized Objects

D.8.1 Domain Objects

D.8.1.1 Means of Production Domain

The name of any Means of Production data domain shall begin with the six character prefix "N_MOP_", complemented by a user or implementer defined means of production identifier, and augmented by a user or implementer defined identifier of the subordinated objects. Parking Locations, Pallets, Partitions, Fixtures, and Parts are subordinated objects of this domain. The Means of Production data domain is detailed in clause 8.1.5 of the main part of this standard.

D.8.3 Named Variable Objects

D.8.3.2 NC Domain Specific Named Variable Objects

D.8.3.2.1 Parking Location Object

The name of each Parking Location object shall begin with the prefix "N_ParkLoc_", complemented by a user or implementer defined parking location identifier.

```
OBJECT: Named Variable
Key Attribute:  Variable Name = domain-specific {
                                    domain ID        "N_MOP_...",
                                    item ID          "N_ParkLoc_... }
     Attribute:    MMS Deletable                 = FALSE
     Attribute:    Type Description              = structure {
        components {
          {component Name = "ParkLocType",
           component Type = visible-string },
          {component Name = "ParkLocStates",
           component Type = unsigned 8      -- 0  EMPTY,
                                            -- 1  not FULL,
                                            -- 2  FULL
                                           },
          {component Name = "ParkLocObjects",
           component Type = array {
                numberOfElements,      -- number of object's
                elementType = visible-string -32 } }
        } }
     Attribute:    Access Method                 = locally defined
```

D.8.3.2.2 N_Partition_ Partition Object

The name of each Partition object shall begin with the prefix "N_Partition_", complemented by a user or implementer defined partition identifier.

OBJECT: Named Variable

```
Key Attribute:  Variable Name = domain-specific {
                                   domain ID      " N_MOP_...",
                                   item ID        " N_Partition_..." }

     Attribute:    MMS Deletable                  = FALSE
     Attribute:    Type Description               = structure {
        components {
          {component Name = "PartitionStates",
          component Type = unsigned 8     - 0  EMPTY,
                                          - 1  RESERVED,
                                          - 2  OCCUPIED
                                          },
          {component Name = "ObjectPresent",
           component Type = visible-string -32 },
          {component Name = "PartitionLocation",
           component Type = visible-string -32 },
          {component Name = "OffsetReference",
           component Type = visible-string -32 },
          {component Name = "PartitionOffsets",
           component Type = structure {
                    components {
                          {component Name = "ARotation",
                          component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                          {component Name = "BRotation",
                           component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                          {component Name = "CRotation",
                           component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                          {component Name = "XTranslation",
                           component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                          {component Name = "YTranslation",
                           component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                          {component Name = "ZTranslation",
                           component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } }
                     } } } } }
     Attribute:    Access Method                  = locally defined
```

OTE - For correct offset definition, transformations shall be applied in the following order: rotation shall precede any translation, applying ler rotations, first about the X axis, then about the transformed Y, then the transformed Z axis. Translations may follow in any order.

D.8.3.2.3 Pallet Object

The name of each Pallet object shall begin with the prefix "N_Pallet_", complemented by a user or implementer defined pallet identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name = domain-specific {
                                   domain ID        "N_MOP_...",
                                   item ID          "N_Pallet_..." }
     Attribute:    MMS Deletable                    = FALSE
     Attribute:    Type Description                 = structure {
        components {
          {component Name = "PalletType",
           component Type = visible-string },
          {component Name = "PalletState",
           component Type = unsigned 8       – 0  EMPTY
                                             – 1
                                             – 2
                                             },
          {component Name = "PalletObjects",
           component Type = array {
                                numberOfElements,   – number of objects
                                elementType = visible-string -32 } },
          {component Name = "PalletLocation",
           component Type = visible-string -32 },
          {component Name = "OffsetReference",
           component Type = visible-string -32 },
          {component Name = "PalletOffset",
           component Type = structure {
                              components {
                                  {component Name = "ARotation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } },
                                  {component Name = "BRotation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } },
                                  {component Name = "CRotation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } },
                                  {component Name = "XTranslation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } },
                                  {component Name = "YTranslation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } },
                                  {component Name = "ZTranslation",
                                   component Type = floating-point {
                                                  format-width 32,
                                                  exponent-width 8 } }
                          } } } } }
   Attribute:    Access Method                      = locally defined
```

NOTE - For correct offset definition, transformations shall be applied in the following order: rotation shall precede any translation, applying Euler rotations, first about the X axis, then about the transformed Y, then the transformed Z axis. Translations may follow in any order.

D.8.3.2.4 Fixture Object

The name of each Fixture object shall begin with the prefix "N_Fixture_", complemented by a user or implementer defined fixture identifier.

OBJECT: Named Variable

```
Key Attribute:   Variable Name = domain-specific {
                                 domain ID       "N_MOP_..."
                                 item ID         "N_Fixture_..." }
      Attribute:   MMS Deletable                  = FALSE
      Attribute:   Type Description               = structure {
        components {
          {component Name = "FixtureType",
           component Type = visible-string },
          {component Name = "FixtureStates",
           component Type = unsigned 8 },  -- user defined state list
          {component Name = "ObjectPresent",
           component Type = visible-string -32 },
          {component Name = "FixtureLocation",
           component Type = visible-string -32 },
          {component Name = "OffsetReference",
           component Type = visible-string -32 },
          {component Name = "FixtureOffsets",
           component Type = structure {
                     components {
                             {component Name = "ARotation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                             {component Name = "BRotation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                             {component Name = "CRotation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                             {component Name = "XTranslation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                             {component Name = "YTranslation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                             {component Name = "ZTranslation",
                              component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } }
                     } } } } }
      Attribute:   Access Method                  = locally defined
```

NOTE - For correct offset definition, transformations shall be applied in the following order: rotation shall precede any translation, applying Euler rotations, first about the X axis, then about the transformed Y, then the transformed Z axis. Translations may follow in any order.

D.8.3.2.5 Part Object

The name of each Part object shall begin with the prefix "N_Part_", complemented by a user or implementer defined part identifier.

OBJECT: Named Variable

```
Key Attribute: Variable Name   = domain-specific {
                                      domain ID        "N_MOP_...",
                                      item ID          "N_Part_..." }
     Attribute:     MMS Deletable                  = FALSE
     Attribute:     Type Description               = structure {
        components {
          {component Name = "PartType",
           component Type = visible-string },
          {component Name = "PartStates",
           component Type = unsigned 8         -- 0  Raw
                                               -- 1  Modified
                                               -- 2  Finished
                                               -- 3  Scrapped
                                               -- 4  Rework
                                              },
          {component Name = "PartLocation",
           component Type = visible-string -32 },
          {component Name = "OffsetReference",
           component Type = visible-string -32 },
          {component Name = "PartOffsets",
           component Type = structure {
                     components {
                            {component Name = "ARotation",
                             component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                            {component Name = "BRotation",
                             component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                            {component Name = "CRotation",
                             component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                            {component Name = "XTranslation",
                             component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                            {component Name = "YTranslation",
                             component Type = floating-point {
                                        format-width 32,
                                        exponent-width 8 } },
                           {component Name = "ZTranslation",
                            component Type = floating-point {
                                       format-width 32,
                                       exponent-width 8 } }
                     } } } },
                {component Name = "PartData",
                 component Type = }    -- user defined
            } }
     Attribute:     Access Method                  = locally defined
```

NOTE - For correct offset definition, transformations shall be applied in the following order: rotation shall precede any translation, applying Euler rotations, first about the X axis, then about the transformed Y, then the transformed Z axis. Translations may follow in any order.

D.9 Conformance

D.9.4 PICS for standardized objects

Domain specific Named Variable Objects
N_ParkLoc_...
N_Partition_...
N_Pallet_...
N_Fixture_...
N_Part_...

Annex E
Application Examples

This informative annex contains NC specific examples.

E.1 Example TURNING CENTRE

E.1.1 Application Description

This example maps a 4 axis horizontal turning centre, also referenced as a dual turret slant bed lathe, to the NC VMD Model. Such a turning centre contains two independent, by-passing turrets which are positioned by controlling processes running separate part programs. The system may be configured for execution of these part programs to be synchronized, thus allowing the programs to be started and stopped in unison, or the turrets may be controlled individually in an asynchronous manner. Figure 36 illustrates the basic components of the turning centre. Associated with each turret are sets of tooling data describing the turret setup, part offsets, device switches, status, and process information. In preparation for operation, user programs are selected and assigned to each of the two controlling processes. To start one or both programs a single synchronized cycle start may be used or individual path cycle start push buttons may be selected. If the paths are synchronized, an alarm condition or feedhold request should halt both programs. Otherwise the programs may be started or stopped independently. Once running the user programs may re-synchronize with one another at various points as they machine the part. The numerical controller may manage the turning centre as a single system or as a collection of devices. Auxiliary mechanisms may also be integrated to perform automatic tool insert and part changes. Switches are provided to enable the control features and status to reflect the results of the operator action.

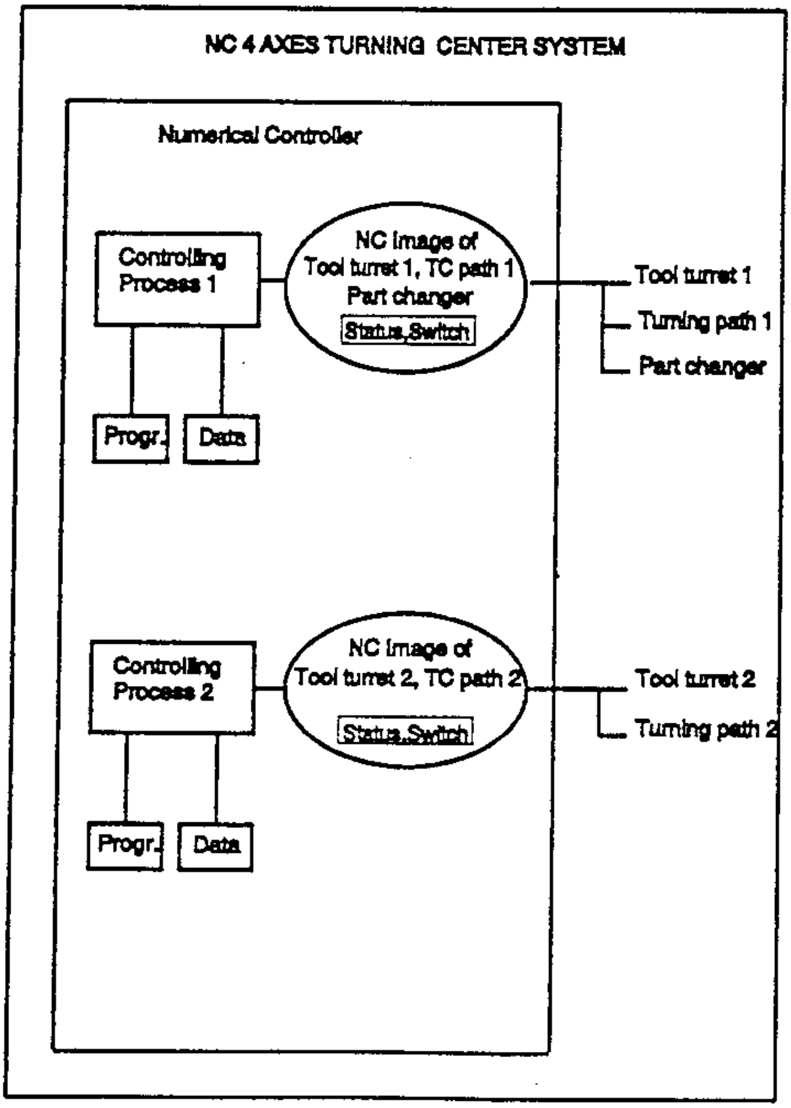


Figure 36 - 4 Axes Turning Centre

E.1.2 The Turning Centre NC VMD

To create a Virtual Manufacturing Device to represent the turning centre, the components which compose the system are mapped to the logical MMS objects which may then be manipulated remotely through use of the MMS services. This section demonstrates such a mapping in accordance with this part of ISO/IEC 9506. Figure 37 illustrates the mapping of turning centre objects to NC MMS objects.

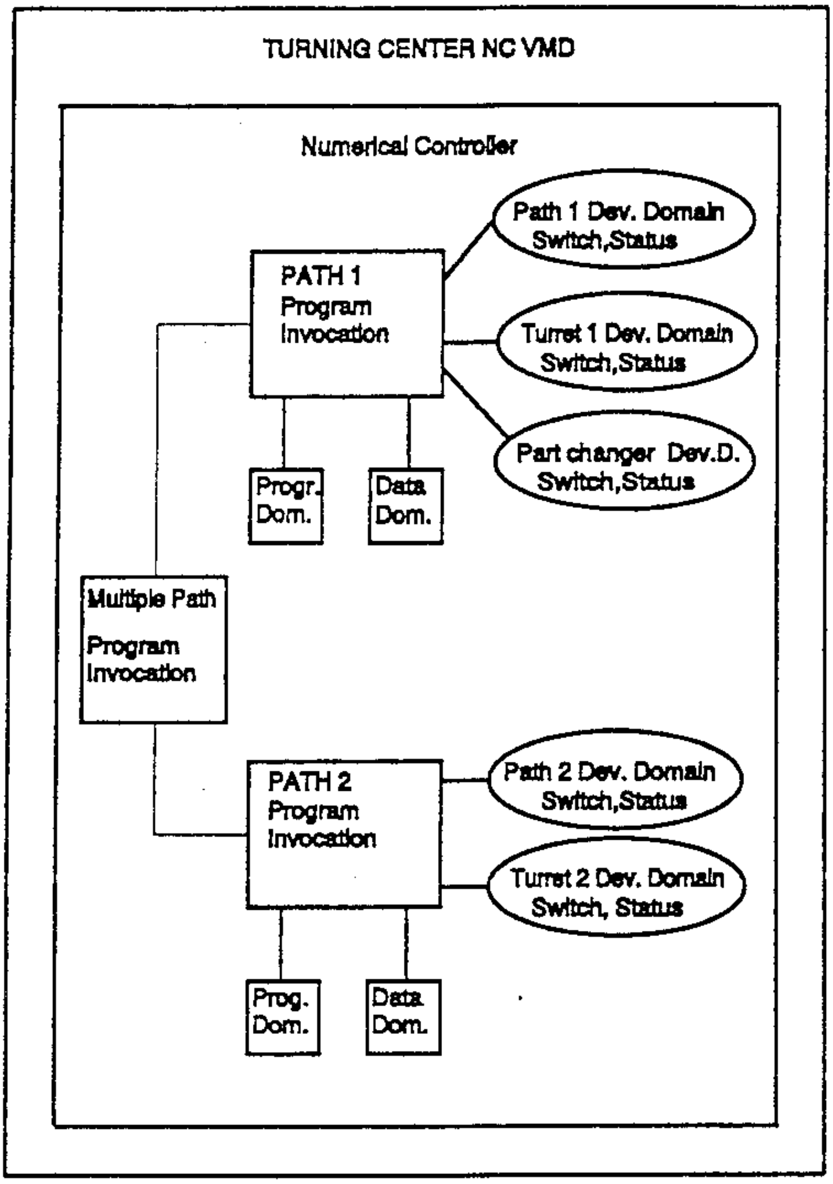

TURNING CENTER NC VMD
Numerical Controller
PATH 1 Program Invocation
Path 1 Dev. Domain Switch,Status
Turret 1 Dev. Domain Switch,Status
Part changer Dev.D. Switch,Status
Progr. Dom.
Data Dom.
Multiple Path Program Invocation
PATH 2 Program Invocation
Path 2 Dev. Domain Switch,Status
Turret 2 Dev. Domain Switch, Status
Prog. Dom.
Data Dom.

Figure 37 - Turning Centre NC VMD

E.1.2.1 Device Mapping

Each of the physical devices integrated into the turning centre system are represented by a Device Domain object. These objects are static and may not be deleted using MMS. The domains may then be bound to controlling processes which utilize their resources. Each Device Domain object should contain two sets of domain subordinate Named Variable objects. The first set of Named Variables represent the switches which are used to activate and align the devices. These may be written by the NC-CS user to set the devices to their ready state. The second set of variables reflect the status and feedback components of the device and may be read or may be sent in an Information Report as a result of a state change. In this example the devices which comprise the axis drives, tool turrets, and part changers are each mapped to a Device Domain.

E.1.2.2 Program Mapping

The user programs which are executed by the controlling processes are mapped to NC program domains. Depending on the local store capacity of the NC, the program domains may be downloaded as needed then deleted using the MMS services, or may be permanently stored at the NC. These program domains will then be allocated for use by the turning centre path controlling processes.

E.1.2.3 Controlling Process Mapping

The controlling processes which execute the turning centre user programs are represented by MMS Program Invocation objects. The program invocations are defined to generate a binding to the domains which represent the physical and logical resources of the system. Some resources may be sharable, such as a part changer domain while others are dedicated to a single program invocation. Because the four axes of the turning centre may either be operated in pairs or all four as a single entity, and the devices and program are allocated to specific paths, each of the paths' controlling processes are mapped to a program invocation. These program invocations may be statically or dynamically defined depending on whether or not the domain bindings shall be alterable. When the paths are to be run independently the MMS Program Invocation services are used to start, stop, and resume each of the processes separately. To support the case where the NC synchronizes execution of the programs of both paths, a third Program Invocation is created to operate both of them together. When a Start service is issued for such multiple path Program Invocation both Controlling Processes are started as a local action and their continued operation is reflected in the state of the multiple path Program Invocation. The multiple path Program Invocation becomes unrunnable when either of the individual programs are unavailable. When the Program Invocations are created a domain is also created for each which contains the process information relevant to the controlling process. The information in this domain is contained in domain subordinate Named Variables. These variables may contain active feedrate, spindle speed, offsets and other data which changes as a process runs. The information in the multiple path domain should contain data merged from the two original, individual process information domains.

E.1.2.4 Status And Alarm Mapping

Status and alarm reporting for the turning centre is accomplished by defining Event Condition objects for detectable conditions. Event Actions are defined to convey the information to the NC-CS user when detected and Event Enrollments to pair the actions to the conditions. Acknowledgment is required to guarantee that the NC-CS user is notified.

E.1.2.5 Use of Objects for the Turning Centre Example

This describes the method of controlling the NC system shown in figures 36 and 37 from the supervisory computer actually using MMS.

E.1.2.5.1

The following objects shall be assumed to exist in the NC system from the time of power up.

Program domains	- N_PRG_01000 - N_PRG_02000
Device domains	- N_DEV_P1 (Domain for turning path 1) - N_DEV_P2 (Domain for turning path 2) - N_DEV_T1 (Domain for turret 1) - N_DEV_T2 (Domain for turret 2) - N_DEV_PART (Domain for part changer)
Variables	- N_DEV_STATE The variable which shows the device state. Since this is a Domain Specific variable, each of the device domains has the variable with the same name.
Event condition	- N_SERVO This is used to activate the trigger when a servo excess error occurred.

E.1.2.5.2

In addition, the NC System creates the following objects as required.

Program Invocation	- PATH_1 (for controlling the Controlling Process 1) - PATH_2 (for controlling the Controlling Process 2)
Process Information domain	- N_PID_P1 (Domain which shows the state of the Controlling Process 1) - N_PID_P2 (Domain which shows the state of the Controlling Process 2)
Variable	- N_PRINF_M30 The variable which shows that the Controlling Process has executed M 30 (see ISO 6983). Since this is a Domain Specific variable, each above mentioned two process information domains has the variable with the same name.

E.1.2.5.3

The supervisory computer (NC-CS user) controls the NC system according to the following procedure:

NC-CS user - Initiate-Request

NC system - Initiate-Response

(The association has now been established between the NC-CS user and the NC system).

NC-CS user - Read-Request
The state of the device TURRET 1 may be found by reading the variable N_DEV_STATE of the domain name N_DEV_T1.

NC system - Read-Response
The value is 1 (POWER OFF), 2 (READY), OR 3 (NOT READY).

All the devices are confirmed to be in the READY state by performing Read-Request on TURNING PATH1, TURRET 2, and TURNING PATH2.

NC-CS user - CreateProgramInvocation-Request {
Program Invocation NAME "PATH_1"
List of Domain "N_PRG_01000"."N_DEV_T1" "N_DEV_P1" }

When the NC system receives CreateProgramInvocation-Request, it automatically creates the domain N_PID_P1 for the process information of PATH1, and further defines the variable N_PRINF_M30 in the domain.

NC-CS user - CreateProgramInvocation-Request {
Program Invocation Name "PATH_2"
List of Domain "N_PRG_02000" "N_DEV_T2" "N_DEV_P2" }

NC system - CreateProgramInvocation-Response

When the NC system receives CreateProgramInvocation-Request, it automatically creates the domain N_PID_P2 for the process information of PATH2, and further defines the variable N_PRINF_M30 in the domain.

E.1.2.5.4

At this point, two Program Invocations whose names are PATH1 and PATH2, respectively, have been created. Here, let us try to control these two Program Invocations simultaneously. For this purpose, the additional Program Invocation PATH3 is created.

NC-CS user - CreateProgramInvocation-Request {
Program Invocation Name "PATH3"
List of Domain "N_DEV_PART" }

NC system - CreateProgramInvocation-Response

NC-CS user - Start-Request {
Program Invocation Name "PATH3" }

NC system - Start-Response

E.1.2.5.5

Both PATH 1 and PATH2 have entered the RUNNING state. How PATH1 and PATH2 are related to PATH3 is a local matter. In this state, the feed rate override value in use is the value in LOCAL MODE. This is because the value of the variable N_FRO_ON is false. The feed rate override of Turning Path1 is controlled from the NC-CS user.

NC-CS user - Write-Request
The NC-CS user writes the override value to the variable N_FRO of the device name N_DEV_P1.

NC system - Write-Response

NC-CS user - Write-Request
The NC-CS user makes the variable N_PRO_ON of the device name N_DEV_P1 true.

NC system - Write-Response

E.1.2.5.6

From this point, the value of N_FRO is adopted as the override value.

E.1.2.5.7

The NC system notifies the NC-CS user of a servo excess error.

NC system	- EventNotification-Request { Event Enrollment Name Event Condition Name "N_SERVO" Condition State ACTIVE Acknowledgment Rule ACK_ACTIVE }
NC-CS user	- AcknowledgeEventNotification-Request
NC system	- AcknowledgeEventNotification-Response

E.1.2.5.8

The NC-CS user is notified that M30 was detected in the executing program.

NC system	- InformationReport-Request This system notifies the variable N_PRINF_M30 of the domain name "N_PID_P1".
	- InformationReport-Request This system notifies the variable N_PRINF_M30 of the domain name "N_PID_P2". The NC-CS user may know that the program which has been executed with PATH1 and PATH2 has terminated.

E.1.2.5.9

This concludes a series of processing which controls the NC system.

E.2 Example Milling Centre In A Flexible Manufacturing System

E.2.1 Application Description

This example, illustrated in figure 38, describes a milling centre which is integrated into a flexible manufacturing system. The milling centre consists of three units, a milling machine unit, a tool magazine unit, and a tool loading unit. An automated transport system carries pallets with new parts and tools to the milling centre and removes manufactured parts and used tools from the milling centre. The tool loading unit inserts new tools from a pallet into the tool magazine and removes used tools from the tool magazine to the pallet. During the manufacturing process the tool changer of the milling machine unit also exchanges tools between the tool magazine and the spindle of the milling machine. A NC controls the motion of the milling machine unit and also of the tool loading unit.
The axes of the milling unit and the axes of the tool loading unit are positioned by controlling processes running separate programs.

During the manufacturing process the programs for the different units may be invoked individually. These programs are synchronized in order to avoid a possible collision between the tool changer of the milling unit and the tool changer of the tool loading unit.

Associated with the milling unit or the tool loading unit are sets of pallet data describing the location of the mounted objects, offset data for parts, tooling data, tool magazine data, device switches and status and process information.

When a pallet has reached the working area of the milling machine or the tool loading unit, the NC informs the remote NC-CS User. The NC may obtain information about the pallet (such as the pallet ID) from a code carrier mounted on the pallet.

After the NC has informed the NC-CS User, the NC-CS User downloads data necessary for the manufacturing process of the object(s) mounted on the pallet and starts the process.

When the NC has finished the process, it informs the NC-CS User and the pallet may be transferred to the next station.

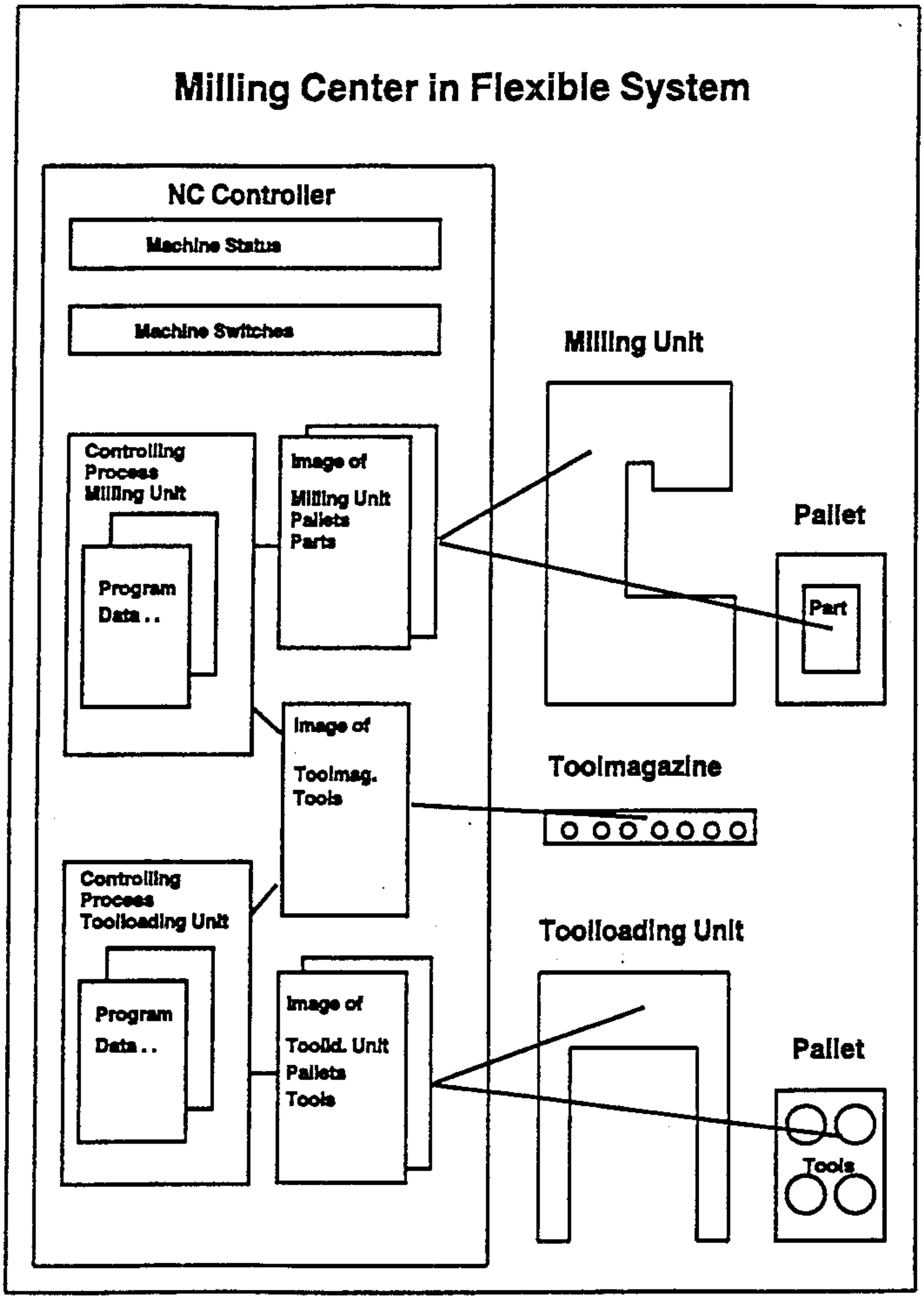


Figure 36 - Milling Centre in Flexible System

E.2.2 The Milling Centre NC VMD

To create a virtual manufacturing device to represent the milling centre, the constituent components of the system are mapped to logical MMS objects which may then be manipulated remotely through use of MMS services. This section demonstrates such a mapping in accordance with the NC companion standard.

E.2.2.1 Device Mapping

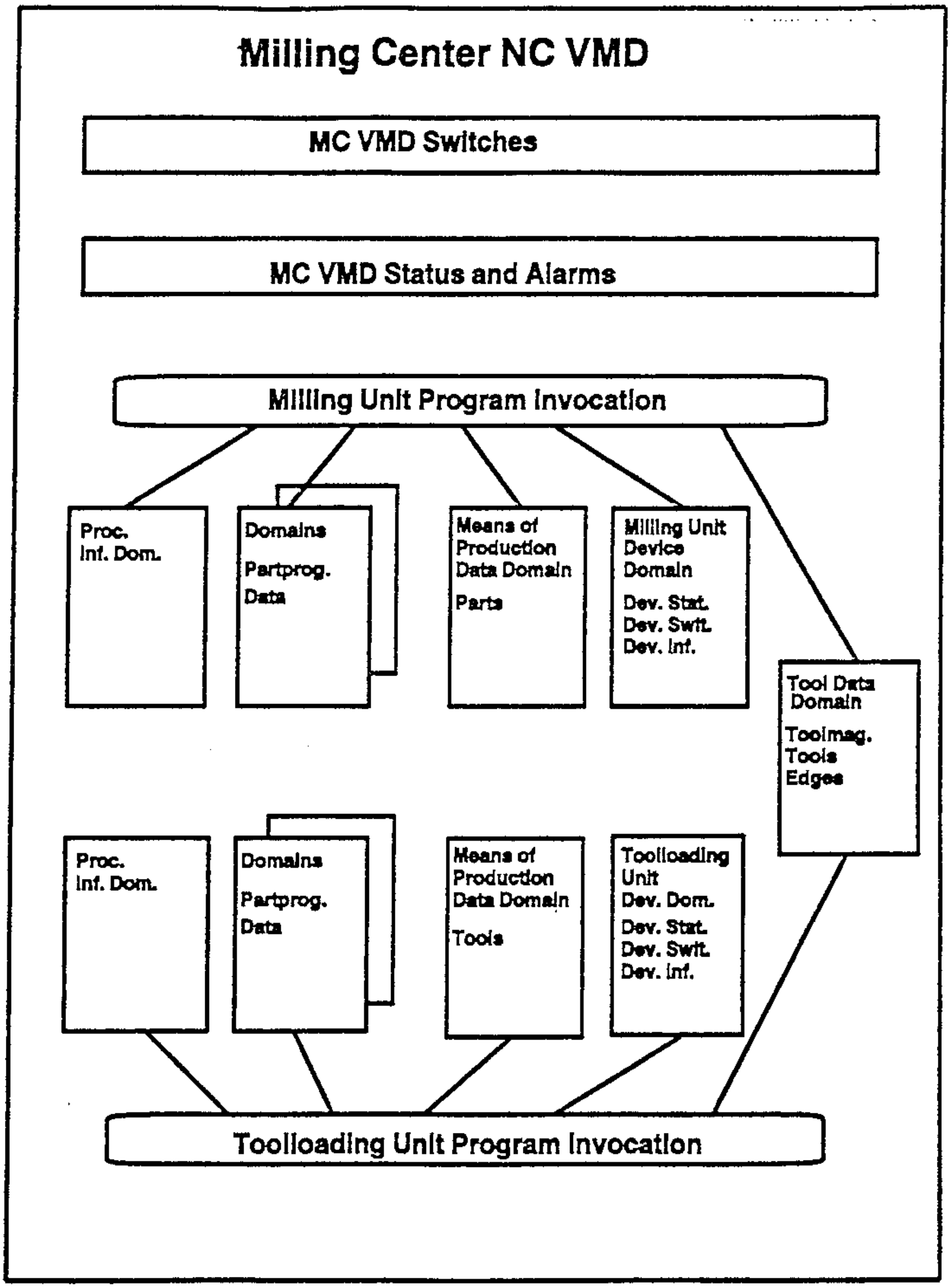
The milling machine unit and the tool loading unit are represented by NC Device Domain objects, which are illustrated in figure 39. In this example the milling unit and the tool loading unit are each mapped to an NC Device Domain which is static and may not be deleted using MMS. These domains may be bound to Controlling Processes which may then utilize their resources. Each Device Domain object contains domain-specific Named Variable objects. A set of Named Variables represents the switches which are used to activate and align the devices. An other set of variables reflects the status of the device and may be read or may be sent in an information report as a result of a state change. A third set of variables reflects the resources of the unit, such as the NC-axes belonging to this unit.

The tool magazine is represented by a Named Variable object. This object is a subordinated object of the tool data domain. The tool data domain contains also sets of Named Variables representing tools, along with their edges, which are located at this tool magazine. The number of the tools may change during the manufacturing process.

For the purpose of updating, the tool data domain may be up- and downloaded to the NC-CS User using MMS services.

Pallets are represented by Named Variable objects. These objects are subordinated objects of the Means Of Production data domain. Parts or tools mounted on a pallet are also represented by Named Variable objects which are subordinated objects of the Means Of Production data domain. This domain is used for up- and downloading of the pallet data. For the tools on the pallet, this domain may contain addition information, such as a list of variables which defines the tools which are to be removed from the tool magazine.

When a pallet reaches the milling centre, the associated Means Of Production data domain may be downloaded as needed, using MMS services. When the manufacturing process associated with the pallet is finished and the pallet is transferred to the next station then the Means Of Production data domain may be uploaded to the NC-CS user.



gure 39 - Milling Centre NC VMD

E.2.2.2 Program Mapping

The programs which are executed by the controlling processes are mapped to NC Program Domains. Depending on the local store capacity of the NC, the program domains may be downloaded as needed, or may be permanently stored in the NC.

These program domains will then be allocated for use by the unit controlling process.

E.2.2.3 Controlling Process Mapping

The controlling processes which execute the milling machine programs and the tool loading unit programs are represented by MMS Program Invocation objects. A Program Invocation establishes a binding to the domains which represent the physical and logical resources of the system.

Some resources may be sharable, such as the tool data domain while others are dedicated to a single program invocation. These program invocations may be statically or dynamically defined depending on whether or not the domain bindings shall be changed. MMS Program Invocation services are used to start, stop and resume the processes.

The synchronization of the milling machine programs and the tool loading programs is a local matter and is not visible to the NC-CS User. When the program invocations are created, a corresponding domain is also created which contains the process information relevant to the controlling process. The information in this domain is contained in domain-specific Named Variables. These variables may contain information such as the actual subroutine, the actual process state and so forth.

E.2.2.4 Alarm Mapping

Monitoring and signaling of alarms for the milling centre is modeled by Event Condition objects. For every alarm of the milling centre which is to be reported to the NC-CS User, an Event Condition object is defined. when the event condition state changes an NC Event Notification is sent to the NC-CS User. The event notification sent to the remote NC-CS User contains information which details the alarm in accordance with the definitions of this standard.

E.2.2.5 Device Information Mapping

Device information data should be mapped to device domain specific Named Variables. These variables may be read using MMS Read Service or may be transferred to the NC-CS User in an Information Report.

In this example the arrival of a pallet at the machine may be mapped on a Device domain specific Named Variable which contains the pallet ID. When a pallet arrives at the milling machine unit the state of this variable changes. As a result of this an information report is sent to the NC-CS User containing this variable. The transfer of the pallet ID from the code carrier of the pallet to this variable is handled locally within the NC.

E.3 Examples of Offsets

This clause describes the use of the objects defined in clauses 8.3.2.2 through 8.3.2.5 with the help of examples.

E.3.1 Example 1: Zero Offset

The NC-CS user (a cell controller, for instance) first establishes a communication connection with the NC system by using the MMS Initiate service. For producing a scheduled part on the system, the NC-CS user first employs the GetNameList service to check if the domains containing the necessary user program(s), tool data, and other means of production are currently in the NC system. If the necessary domains do not exist, the user creates them with the download services (MMS InitiateDownload, DownloadSegment, TerminateDownload services). The workpiece (part blank) is assumed to be mounted on a fixturing device. The user program is written for the part reference system.

The zero offset (for one axis) is the difference between the real axis zero point and the zero point of the user program, in this case the difference between axis zero and workpiece zero. The zero offset for the x and y axes was measured earlier (by a measurement system) and stored within the NC-CS user which now writes the values into the domain-specific variable N_DEV_ExampleMachine. N_ZeroAxes by the MMS Write service. The structure of this variable is the following in pascal/c like notation:

```
N_DEV_ExampleMachine.N_ZeroAxes = RECORD
        on BOOLEAN;                               -- global zero offset on
        zeroOffsets ARRAY[1..4] OF RECORD         -- dev-ExampleMachine-number
                                                  -- of-axes = 4 (PICS)

             nameOfAxis:identifier;
             on: BOOLEAN;                          -- zero offsets for a certain
                                                  -- axis on

             zeroOffsets : ARRAY[1..1] OF RECORD
                                                  -- dev-ExampleMachine-number-
                                                  -- of-zero-offsets = 1 (PICS)
                  on:BOOLEAN ;                    -- value valid
                  value: floating-point;

        END;
    END;
END;
```

with the value (after the write operation)

```
{ TRUE,                              -- zero offset global on
      "x", TRUE,                     -- offsets for "x" on
           TRUE, 10.5,               -- first offset on, value 10.5
      "y", TRUE,                     -- offsets for "y" on
           TRUE, -1.12,              -- first offset on, value -1.12
      "z", FALSE,                    -- no offsets for "z"
           ignored, ignored,         -- first offset ignored, value ignored
      "SillyAxis", FALSE,            -- offsets for "SillyAxis" off
           ignored, ignored          -- first offset ignored, value ignored }
```

Through the MMS CreateProgramInvocation service, the NC-CS user now may create the Program Invocation object ProgramForPart containing the user program and tool data (domains) in the list of domain references. If the NC system is in REMOTE mode, the NC-CS user may start the execution of the program by using the MMS Start service on the above Program Invocation object ProgramForPart.

NOTE - The zero offset structure above may be used only to compensate for linear displacement of the workpiece. For compensation of displacements containing rotary components, a more complex transformation is required, as shown below.

E.3.2 Example 2: Axis Value Transformation

From the original part program, the NC-CS user may want to produce a mirrored part, (such as, for instance, producing a left shoe from the user program for a right shoe). For this purpose, consider the following layout of the workpiece:

```
virtual axis (workpiece) ------------+RIGHT–LEFT----------
```

For producing the mirrored part (called LEFT) with the same part program, the NC-CS user first applies a mirroring primitive (mirroring at the virtual zero), then a translation primitive, for moving the mirrored part to correct location within the work space.

```
virtual axis          ----------+RIGHT-------------
Mirroring M()           --------LEFT+--------------
Translation T()                 ------
                    -----------+-----LEFT----------
```

The NC-CS User writes, with the MMS Write service, the correct primitives into the variable N_DEV_ExampleMachine.N_AxesValTrans. This variable has the following structure (in pascal/c like notation)

```
N_DEV_ExampleMachine.N_AxesValTrans = RECORD
        on BOOLEAN;                                         – Transformations on
        transformations   ARRAY[1..4] OF RECORD
                                                            –  dev-ExampleMachine-
                                                            – number-of-axes- = 4
        nameOfAxis:identifier;                              – name of axis
        on: BOOLEAN;                                     – axis on
        transformations : ARRAY[1..5] OF RECORD
                                                         -- dev-ExampleMachine-
                                                         – number-of-axis-value-
                                                         -- transformations = 5
        on: BOOLEAN;                                     – primitive valid
        primitive: (Translation(0), Mirroring(1), Scaling(2));
        value:floating-point;
     END;
   END;
END;
```

with the value (after the write operation)

```
{ TRUE,                                   -- axis value transformations global on
   "x", TRUE,                             -- transformations for "x" on
      TRUE, 1, ignored,                   – first primitive Mirroring(), no parameter
      TRUE, 0, 4,                         – second primitive Translation(4)
      FALSE, ignored, ignored ,           – no third primitive
      FALSE, ignored, ignored,            – no fourth primitive
      FALSE, ignored, ignored ,           – no fifth primitive
   "y", FALSE,                            – no transformations for "y"
      ignored,ignored,ignored,
      ignored,ignored,ignored,
      ignored,ignored,ignored,
      ignored,ignored,ignored,
     ignored,ignored,ignored,
   "z", FALSE,                            -- no transformations for "z"
      ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored,
   "SillyAxis", FALSE,                    – no transformations for "SillyAxis"
     ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored,
     ignored,ignored,ignored
```

E.3.3 Example 3: Coordinate Transformations/Active Part Coordinate System

In more complex applications, a work station under control of the NC-CS user may use pallets, fixtures, and tools as described in annex D. The offsets (coordinate system transformations) required for any of these objects may be acquired in one of the following ways: One, with the help of a built-in chip, programmed through a measuring system. This way the corresponding objects may be created automatically with the correct offset information, when a pallet comes into the area of the work station. In another scenario the pallets and fixtures do not have the information on board, but the measuring system which measures the offsets transfers the values to the NC-CS user (cell controller) who may use the MMS Write service to write (or the NC requests) these values into the corresponding fields of the appropriate, automatically created objects, which may be structured, for example, with the following hierarchy:

```
Machine_System   (known by the NC)
   PalletObject1 (Pallet_Coordinate_System1)
      FixtureObject1 (Fixture_Coordinate_System1)
         PartObject1 (Part_Coordinate_System1)
         PartObject2 (Part_Coordinate_System2)
         PartObject3 (Part_Coordinate_System3 )
```

```
        FixtureObject2 (Fixture_Coordinate_System2)
            FixtureObject3 (Fixture_Coordinate_System3)
                PartObject4 (Part_Coordinate_System4)
```

n this system of reference, for instance, the program for part 1 (PartObject1) which is written in terms of Part_Coordina-e_System1, resides at the NC-CS user. To inform the NC which coordinate system to use, the NC-CS user writes into the lomain-specific variable N_DEV_ExampleMachine.N_ActProg_CS the (identifier) value PartObject1. Starting from the Machine_System, a known device attribute, together with the hierarchy of the objects, which relates all offsets (coordinate ransformations) from the machine_system to the part coordinate system (the "active program coordinate system"), the NC ystem may now compute the location and orientation of part1, and is ready for the production.

N_DEV_ExampleMachine.N_ActProg_CS = PartObject1

:.3.4 Example 4: Geometry Transformation

From the original part program, the NC-CS user may want to produce a similar but different size part, such as producing a size 6 shoe from a size 5 part program. In this case the second part shall be scaled with a factor 1.2 in all three dimensions. For this case consider the following layout of the workpiece:

```
three dimensional)
art coordinate system ————————+Size5————————
\fter scaling with 1.2 ———————+Size_6———————
ranslation T()                 ————
\fter Translation      ——————+————Size_6———
```

The NC-CS user setups the N_DEV_ExampleMachine.N_GeometryTrans variable with the following primitives: 1. scaling ınd 2. Translation, with a MMS write service. The variable has the following structure (in c/pascal like notation):

```
I_DEV_ExampleMachine.N_GeometryTrans = RECORD
   on BOOLEAN;           – global Transformations on
   transformations: ARRAY[1..10] OF RECORD
                            -- dev-ExampleMachine-number-
                            – number-of-geometric-transformations = 10
     on: BOOLEAN;       – primitive valid
     primitive:  (Translation(0), Rotation(1), Scaling(2),
                  Mirroring(3));
       value1: floating-point;
       value2: floating-point;
       value3: floating-point;
  END;
:ND;
```

vith the following values:

```
 TRUE,                 – Transformation on
   TRUE, 2, 1.2, 1.2, 1.2, -- first primitive, Scaling with 1.2, 1.2, 1.2
   TRUE, 0, 5, 0, 0,    – second primitive, Translation with 5,0,0
   FALSE, ignored,ignored,ignored,ignored,   – no third primitive
   FALSE, ignored,ignored,ignored,ignored,   – no fourth primitive
   FALSE, ignored,ignored,ignored,ignored,   – no fifth primitive
   FALSE, ignored,ignored,ignored,ignored,   – no sixth primitive
   FALSE, ignored,ignored,ignored,ignored,   – no seventh primitive
   FALSE, ignored,ignored,ignored,ignored,   – no eighth primitive
   FALSE, ignored,ignored,ignored,ignored,   – no ninth primitive
   FALSE, ignored,ignored,ignored,ignored    – no tenth primitive }
```

주의

1. 이 보고서는 과학기술처에서 시행한 특정연구개발사업의 연구보고서입니다.

2. 이 보고서 내용을 발표할 때에는 반드시 과학기술처에서 시행한 특정연구개발사업의 연구결과임을 밝혀야 합니다.

3. 국가과학기술 기밀유지에 필요한 내용은 대외적으로 발표 또는 공개하여서는 아니됩니다.