Josef Scheller

Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen

Josef Scheller

Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen

Herausgegeben von
Professor Dr.-Ing. Klaus Feldmann,
Lehrstuhl für
Fertigungsautomatisierung und Produktionssystematik
FAPS

Carl Hanser Verlag München Wien

Als Dissertation genehmigt von der Technischen Fakultät
der Friedrich-Alexander-Universität Erlangen-Nürnberg

Tag der Einreichung:	07. 01. 1991
Tag der Promotion:	04. 03. 1991
Vorsitzender:	Prof. Dr.-Ing. F. Hofmann
Berichterstatter:	Prof. Dr.-Ing. K. Feldmann
	Priv.-Doz. Dr.-Ing. habil S. Keramidis

Die Deutschen Bibliothek - CIP-Einheitsaufnahme

Scheller, Josef:
Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen / Josef Scheller - München; Wien; Hanser 1991
(Fertigungstechnik - Erlangen; 18)
Zugl.: Erlangen, Nürnberg, Univ., Diss.
ISBN 3-446-16454-5
NE: GT

Dieses Werk ist urheberrechtlich geschützt.
Alle Rechte, auch die der Übersetzung, des Nachdrucks und der Vervielfältigung des Buches oder Teilen daraus, vorbehalten.
Kein Teil des Werkes darf ohne schriftliche Genehmigung des Verlages in irgendeiner Form (Fotokopie, Mikrofilm oder ein anderes Verfahren), auch nicht für Zwecke der Unterrichtsgestaltung - mit Ausnahme der in den §§ 53, 54 URG ausdrücklich genannten Sonderfälle - , reproduziert oder unter Verwendung elektronischer Systeme verarbeitet, vervielfältigt oder verbreitet werden.

©Carl Hanser Verlag München, Wien 1991
Herstellung: Copy Center 2000, Erlangen-Eltersdorf
Printed in Germany

Vorwort

Die vorliegende Dissertation entstand während meiner Tätigkeit als wissenschaftlicher Mitarbeiter am Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik der Universität Erlangen - Nürnberg.

Herrn Professor Dr. Ing. K. Feldmann, dem Leiter dieses Lehrstuhls am Institut für Fertigungstechnik, danke ich für die wohlwollende Förderung bei der Durchführung dieser Arbeit und für die in praktischen und organisatorischen Bereichen gewonnenen Erfahrungen.

Herrn Privatdozent Dr. Ing. habil. S. Keramidis danke ich für die Übernahme des Korreferates, für die fachlichen Anregungen und Verbesserungsvorschläge. Stellvertretend für alle Mitarbeiter des Instituts für mathematische Maschinen und Datenverarbeitung möchte ich meinen Dank Herrn Professor Dr. Ing. F. Hofmann und Herrn Professor Dr. P. Mertens für die Unterstützung aus der Informatik und Wirtschaftsinformatik sowie für die Zusammenarbeit in unterschiedlichen Forschungsprojekten aussprechen. Gerade diese Integration von Informatik- und Fertigungsaspekten haben die vorliegende Arbeit geprägt.

Dieses Buch leistet einen Beitrag zur Durchdringung der Komplexität heutiger Softwaresysteme bei dispositiven prozeßnahen Steuerungsaufgaben. Die Basisentwicklungen zu einer systematischen und gezielten Softwarekonstruktion wurden sowohl im theoretischen Bereich als auch in der praktischen Umsetzung von meinen Kollegen Herrn Dipl. Ing. R. Klotzbücher und Herrn Dipl. Ing. E. Sommer mitgetragen. Die Kernpunkte dieser Arbeiten liegen auf Aspekten der Materialflußsteuerung bzw. auf der Steuerungssystematik im Gerätebereich; besonderer Dank sei somit diesen beiden für ihren Beitrag zu den entworfenen Konstrukten und für die fruchtbaren Diskussionen gesagt.

Ferner gilt mein Dank allen Kolleginnen und Kollegen - insbesondere der Steuerungs- und Sensorikgruppe -, Studenten und wissenschaftlichen Hilfskräften, die durch ihre Unterstützung zum Gelingen dieser Arbeit - auch in bezug auf Textverarbeitung und Bildgestaltung - beigetragen haben. Einen weiteren Dank möchte ich Herrn Dipl. Ing. H. Reichel und Herrn Dipl. Ing. M. Steber für die Weitergabe ihres maschinenbaulichen und fertigungstechnischen Wissens und ihrer Bereitschaft, im Gegenzug Wissen aus der Informatik in die heutige Fertigungswelt einzubringen, aussprechen. Ein letztes, aber sicherlich ein sehr herzliches Dankeschön möchte ich all jenen Zeitgenossen zurufen, die meinen Blick auch auf andere Interessensgebiete lenkten und gerade hierdurch eine positive Wirkung auf die Durchführung dieser Dissertation ausübten.

Scheller Josef

Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen

Inhaltsverzeichnis

1	Einleitung	1
2	Der Zellengedanke innerhalb der Montageautomatisierung	3
2.1	Abgrenzung und Einordnung flexibler Montagezellen	3
2.2	Problematik der Gestaltung von Steuerungssystemen	6
	2.2.1 Zielsetzungen von Zellensteuerungen	6
	2.2.2 Zellensteuerung mit neuem Profil	7
	2.2.3 Gestaltung von Software	9
2.3	Stand der Technik und Entwicklungstendenzen	10
	2.3.1 Aspekte von Verfahrensketten	10
	2.3.2 Hierarchische Steuerungskonzepte	13
	2.3.3 Normierungs- und Standardisierungsbestrebungen	15
	2.3.4 Qualitätssicherung der Software	18
2.4	Zielsetzung und Vorgehensweise	20
3	Aufgabenorientierte Analyse	23
3.1	Flexibilitätsaspekte in einer Zellensteuerung	24
	3.1.1 Konkretisierung des Begriffs "Flexibilität"	24
	3.1.2 Ausgewählte Aufgabenspektren	25
3.2	Die Arbeitsplanung im CIM Verbund	28
	3.2.1 Arbeitsplaninformationen	29
	3.2.2 Automatisierung der Arbeitsplanung	30
	3.2.3 Strukturen für Zellenarbeitspläne	31
3.3	Betriebsdatenerfassung	32
3.4	Anforderungen an Softwaresysteme	34
	3.4.1 Datenhaltung in Montagesystemen	34
	3.4.2 Betriebssysteme	35
	3.4.3 Kommunikation	36
	3.4.4 Initialisierung und Wiederanlauf	38
	3.4.5 Softwaremodule	39

4	Architekturprinzipien zur Entwicklung der Anwendersoftware	41
4.1	Grundgedanken	41
	4.1.1 Grundlagen der Modellierung	42
	4.1.2 Modell für asynchrone Prozeβsysteme	44
	4.1.3 Kriterien für effiziente Steuerungsstrukturen	50
4.2	Abstraktion mittels hierarchischem Aufbau	52
	4.2.1 Grundlagen einer Abstraktion	52
	4.2.2 Hierarchisches Architekturmodell	53
4.3	Zerlegung der Steuerungssoftware in Komponenten	58
	4.3.1 Modellierungsebenen und mögliche Ansichten	58
	4.3.2 Der Modulbegriff als Basis der Zerlegung	61
	4.3.3 Ermittlung von Komponenten in einem System	63
4.4	Prinzip des Regelkreises in der Zelle	67
	4.4.1 Rückmeldungen für Planung und Steuerung	67
	4.4.2 Der Steuerungs-/BDE-Regelkreis in der Montagezelle	71
	4.4.3 Visualisierungsaspekte	73
4.5	Konzeption einer Arbeitsplanstruktur für Montagezellen	74
	4.5.1 Grundgedanken	74
	4.5.2 Struktur von Arbeitsplänen in einer Montagezelle	75
4.6	Initialisierungsmechanismen,Unterbrechungen, Diagnose	78
	4.6.1 Unterbrechungen eines asynchronen Prozeβsystems	78
	4.6.2 Wiederaufnahme nach geplanter Unterbrechung	83
	4.6.3 Wiederaufnahme nach Blockierung des Prozeβsystems	86
4.7	Anbindung an übergeordnete Systeme	91
5	Mechanismen zur Unterstützung bei der Realisierung	94
5.1	Relationale Datenbanken	94
	5.1.1 Prozeβnahe Anwendung	95
	5.1.2 Relationale Datenmodelle zur Strukturierung	95
5.2	Einsatz und Vorteile von UNIX-Systemen	98
	5.2.1 Aufbau der Systemsoftware	99
	5.2.2 Echtzeit - Aufgaben	99
5.3	Strukturen von Kommunikationsmedien	100
5.4	Verwendung der Softwaretechnologie	101

6	Informationsstrukturen der Abstraktionsebenen	105
6.1	Relevante Datenbereiche	105
	6.1.1 Stammdaten und Arbeitsplandaten	105
	6.1.2 Zustands-, Auftrags-, Materialdaten	108
6.2	Methoden zur Darstellung der Informationen	110
	6.2.1 Knotentypenmethode zur Darstellung der Produktstruktur	110
	6.2.2 Aufbau und Struktur von Rahmenarbeitsplänen	112
	6.2.3 Freiheitsgrade : Station und Ablauf	116
	6.2.4 Die Reduzierung des Strukturgraphen	118
	6.2.5 Eingliederung in das Ebenenmodell	120
6.3	Ebene 6 - Abwicklung von Zellaufträgen	122
6.4	Ebene 5 - Handling von Arbeitsgängen	123
	6.4.1 Zustandsdaten auf Arbeitsgangebene	125
	6.4.2 Arbeitsgangauswahl	126
	6.4.3 Modularisierungsaspekte am Beispiel der Koordinierung	132
6.5	Ebene 4 - Abarbeitung von Arbeitsschritten	134
7	Anwendungen und Realisierungserfahrungen	140
7.1	Doppelroboterzelle	140
7.2	Entwicklung der Software	144
	7.2.1 Verbindung Funktionshierarchie - Softwaretechnologie	144
	7.2.2 Integration von Arbeitsplan und Zustandsdaten	148
	7.2.3 Hilfsmittel zum Entwurf und zur Implementierung	149
7.3	Darstellung der Zellensteuerung als Prozeβsystem	150
	7.3.1 Informelle Beschreibung	150
	7.3.2 Spezifikation des Prozeβdatentyps 'Materialdisposition'	152
7.4	Unterstützung durch relationale Datenbanken	156
7.5	Anwendung im Hinblick auf methodisches Vorgehen	160
	7.5.1 Durchgängigkeit von Spezifikation und Implementierung	162
	7.5.2 Wiederverwendungssaspekte am Beispiel Peripherie	169
8	Zusammenfassung	172
9	Literatur	174

1 Einleitung

Die derzeitige Situation in der Produktionstechnik ist durch eine zunehmende Integration der verschiedenen Produktionsstufen gekennzeichnet. Im Mittelpunkt steht in Zukunft eine mehr und mehr gesamtheitliche Betrachtungsweise der Produktionsabläufe, d.h. die Rechnerunterstützung muß dazu beitragen, die operative Handlungsfähigkeit bei gestiegenen Flexibilitätsanforderungen in den Teilbereichen Fertigung, Montage und Prüfung zu sichern /20,21,30,67/.

Bei der schnellen Entwicklung von Steuerungsgeräten hat jedoch nur ein wachsender Leistungsumfang, kaum aber die Festlegung geeigneter Funktionalitäten für die Montagesteuerung Berücksichtigung gefunden. Insbesondere im prozeßnahen Bereich der Montage wurden die Steuerungsfunktionen speziell auf die jeweilige Anwendung abgestimmt und angepaßt /56/.

Zur Bewältigung der vielfältigen Anforderungen an die Produktion reichen diese Steuerungsphilosophien nicht mehr aus. Deshalb sind neue Strategien zu überlegen, um in einer rechnergeführten Fabrik die komplexe Steuerungstechnik beherrschen zu können. Eine Verbesserung dieser Situation kann durch den Einsatz von Zellenrechnern erreicht werden, die differenzierte Aufgaben von der direkten Prozeß- und der übergeordneten Werkstattsteuerung übernehmen und Verbindungen zu diesen beiden Bereichen herstellen /79/. Der Zellengedanke stellt somit einen wesentlichen Schritt in Richtung Konzeption flexibler Systeme dar.

Im Mittelpunkt der vorgenommenen Betrachtungen steht innerhalb einer hierarchischen Steuerungsstruktur die flexible Montagezelle (FMZ). Für diesen Teilbereich des CAM Gebildes (Computer Aided Manufacturing) werden Entwicklungen aufgezeigt, die sowohl zur Durchdringung dieses Problemfeldes beitragen als auch eine Grundlage zur Verständigung sowie zur Zusammenarbeit unterschiedlicher Systembeteiligter aus verschiedenen Disziplinen bilden.

Zur Überwindung fehlender Strukturierungsprinzipien in der Softwareentwicklung (sogenannte 'Softwarekrise') wurden neue Sprachansätze und Werkzeugunterstützungen vorgeschlagen und realisiert. Zur Lösung anstehender Probleme bildet sich neben dem Einsatz geeigneter Methoden in allen Phasen der Softwareentwicklung sowohl zur frühzeitigen Fehlererkennung und Behebung als auch zur späteren Aktualisierung in der

Produktionsinformatik die Strukturierung der Aufgabenbereiche als neuer besonderer Schwerpunkt heraus /106/.

Denn ohne Verbindungsstrukturen von Fachgebietswissen und Softwarekenntnissen zeigen sich die Grenzen bisher entstandener Softwareentwicklungsmethoden. Daher ist es folgerichtig und von Fachleuten mehr und mehr bekräftigt, daß die Beschäftigung mit CASE-Systemen (Computer Aided Software Engineering) nicht das Problem an sich löst /44/. Vielmehr sollten die meist heterogenen Anforderungen der mit dem zu entwickelnden System in Verbindung stehenden Menschen (Manager, Entwickler, Benutzer, etc.) auch in den einzelnen Fachgebietsanforderungen aufeinander abgestimmt werden /22, 91, 109/.

Die Aufgabe im genannten Problembereich besteht also in der Schaffung einer Basis, die bei der Gestaltung von Softwaresystemen für Montagezellenrechner zur Effizienzsteigerung herangezogen werden kann. In dieser Arbeit wird gezeigt, wie unter Multitaskingsystemen mit Unterstützung von konkreten Architekturprinzipien, Modellierungsmethoden und Beschreibungsmechanismen jeweilige Applikationssoftware entwickelt werden kann.

Hierdurch können wiederverwendbare Lösungsansätze aufgebaut und jeweils individuelle - für unterschiedliche Betrachter nicht mehr überschaubare - Lösungswege vermieden werden. Die Beachtung des aufgestellten Rahmenwerkes ermöglicht somit innerhalb der Entwicklung von Steuerungen im Umfeld der Montage das Gewinnen von Systemstrukturen und erleichtert zugleich die Nutzung bestehender Erfahrungen.

Die Integration von Regelsystemen - Auswerten von Ist-Daten und Generierung neuer Solldaten innerhalb der Zellenrechnersoftware - und ein systematisches ingenieurmäßiges Vorgehen durch Modularisierung und abstrakte Datentypen tragen hierbei nicht nur zur Vereinfachung und Lösung vorgegebener komplexer Automatisierungsaufgaben in der Montage bei, sondern helfen auch bei der Bewältigung neuer Aufgaben durch veränderte Anforderungen.

2 Der Zellengedanke innerhalb der Montageautomatisierung

2.1 Abgrenzung und Einordnung flexibler Montagezellen

Im Bereich der flexiblen Fertigung und Montage haben sich bezüglich des Material-, Werkzeug- und Informationsflusses entkoppelte Zellen als Strukturierungsprinzip durchgesetzt /69/. In einer Montagezelle werden eine oder mehrere Stationen mit automatischen Handhabungs-, Montage- und Prüfgeräten als Kernpunkte bezeichnet. Aus Gründen der Flexibilität sind dies meist NC-Achsen bzw. Roboter mit bis zu sechs Freiheitsgraden.

Aber erst die Integration eines Materialflußsystems (keine feste Verkettungsfolge im Vergleich zu Montagelinien) und die eines zentralen Rechners für die organisatorischen und technischen Steuerungs- und Überwachungsaufgaben (automatischer Informationsfluß und autarke Abarbeitung der Vorgaben von übergeordneten Instanzen) rechtfertigen die Verwendung dieses Begriffes (Bild 1).

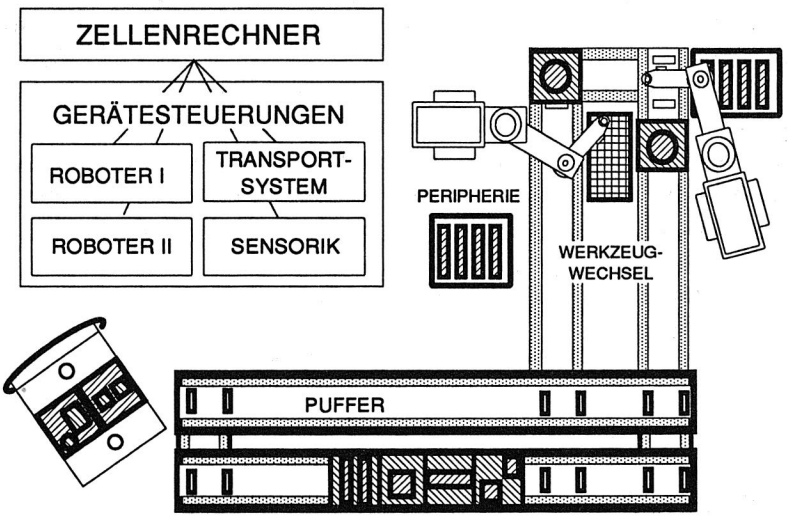


Bild 1: Komponenten und Aufbau flexibler Montagezellen

In einer Montagezelle wird aus Bauteilen bzw. aus Baugruppen eine Vormontagebaugruppe des Endproduktes bzw. dieses selbst zusammengesetzt. Eine Station führt eine oder mehrere Fügeoperationen bzw. Prüfoperationen samt zugehörigen Hilfs- und Handhabungsoperationen aus. In ihrem Peripheriebereich befinden sich neben Ablageplätzen für Werkstücke, Werkzeuge (z.B. Greifermagazine) und Verbrauchsmaterial auch Montageplätze. Außerdem hat jede Station mindestens einen Übergabeplatz, auf dem Werkstücke an ein Transportsystem oder ein Handhabungsgerät übergeben werden können.

Die Arbeitsinhalte sind meist größer ausgelegt als bei Montagelinien; eine Abgrenzung ist jedoch schwierig, da die Anzahl der verschiedenen Montageschritte und die Handhabung unterschiedlicher Werkstücke durch den Arbeitsraum des Industrieroboters, der Zuführproblematik der Teile und durch Aspekte der Wirtschaftlichkeit und Fehleranfälligkeit, z.B. bei häufigen Greiferwechseln, beschränkt wird. Die Montagezelle selbst kann über definierte Ein- / Ausgänge an ein zellenexternes Materialflußsystem (z.B. 'Fahrerloses Transportsystem') - zur Verbindung zum Lager, zu Fertigungszellen oder zu weiteren Montagezellen - angeschlossen werden.

Zur Abgrenzung gegenüber anderen Arten von Montagesystemen wird die Gliederung nach /9/ herangezogen. Die nach Klassen erfolgte Einteilung (Flexibles Montagemodul, Flexible Montagezelle, Flexible Montagegruppe, Flexible Montagesysteme und Flexible Montagelinien) wird besonders mit Montagegegebenheiten und Montageerfordernissen, d.h. nach der unterschiedlichen Struktur der Erzeugnisse (Baugruppen, Produkte), begründet. Hiernach besteht eine Flexible Montagezelle aus mehreren Flexiblen Montagemodulen (NC-Maschine, Industrieroboter, etc., allgemein als Station bezeichnet) und dem erforderlichen Materialfluß, technischen Steuerungs- und Informationsfluß. Der Einsatz ist im Bereich für mittlere bis hohe Produktvielfalt bei geringem bis mittlerem Jahresbedarf empfehlenswert (vgl. Bild 2).

Anhand dieser Einteilung und von Kriterien, die üblicherweise zur Charakterisierung von Produktionsformen /35/ verwendet werden, kann die Position der Zentrenfertigung zwischen der Organisationsform der Werkstatt- und Fließfertigung dargestellt werden.

Das Produktionsspektrum, das in der FMZ montiert werden kann, ist meist durch eine Bildung von unterschiedlichen Produktfamilien gekennzeichnet. Hierdurch wird eine Zusammenfassung ähnlicher durchzuführender Arbeitsoperationen, Kapazitätsfragen und Rüstfragen ermöglicht /31, 38, 66/.

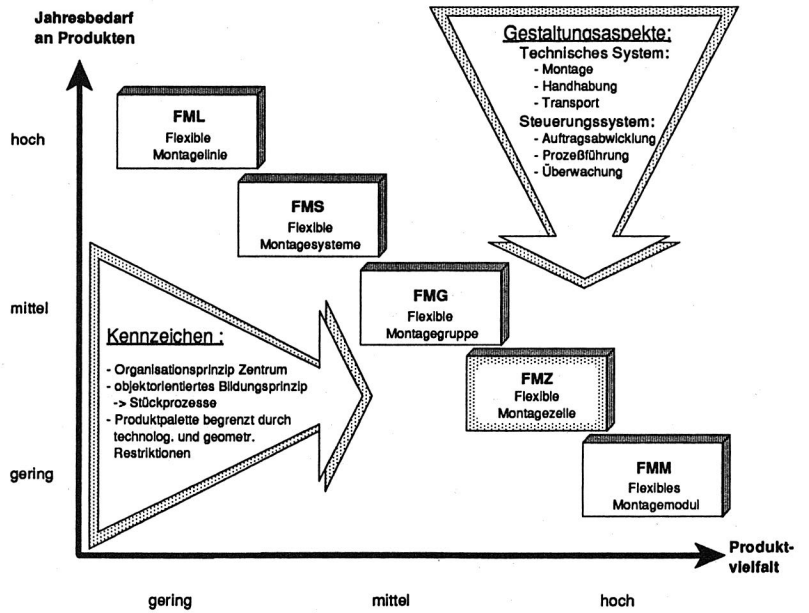


Bild 2: Klassifikation flexibler Montagesysteme

Die Produktionsform Zentrum erfordert im Vergleich zur Fließfertigung einen detaillierteren Arbeitsplan, da hier neben Varianten auch verschiedene Produkte hergestellt werden oder auch beim gleichen Produkt unterschiedliche Herstellungstechnologien zum Einsatz kommen. Der jeweilige Informationsgehalt des Arbeitsplanes, z.B. Freiheiten der Montagereihenfolge, ist sehr unterschiedlich und steht mit Flexibilitätsmerkmalen der Zelle und der Intelligenz der eingesetzten Steuerungen in engem Zusammenhang. Ausprägungen der Zentrenfertigung werden aus der Sicht des Maschinenbaus und der Steuerungstechnik neben der Unterscheidung nach dem Einsatzgebiet Teilefertigung oder Montage durch Kriterien wie automatische Werkstück- und Werkzeugversorgung, Vielseitigkeit (verschiedenartige Produkte) und Anpassungsfähigkeit (Umstellung auf andere Produkte) gekennzeichnet.

Die Montagezelle wird im folgenden nach der in der Literatur bekannten organisatorischen Betrachtungsweise /18/ als die kleinste in sich funktionierende organisatorische Einheit im Betrieb angesehen. Die Arbeitsinhalte eines an eine Zelle gerichteten Auftrags können demnach vollständig in dieser Zelle mit den vorhandenen Betriebsmitteln ausgeführt werden. Die in der Montagezelle stattfindenden Montage-, Transport-, Prüf-, und

Lagervorgänge (im Zusammenhang mit Bereitstellen) werden dann unter einer funktionsorientierten Sichtweise betrachtet. Sie eignet sich sowohl zur Untersuchung und Definition von kleinen Funktionskomplexen, um zu einer detaillierten Untergliederung zu gelangen, als auch zur Verdeutlichung von Zustandsänderungen angesprochener Objekte (Material, Betriebsmittel, Aufträge).

2.2 Problematik der Gestaltung von Steuerungssystemen

2.2.1 Zielsetzungen von Zellensteuerungen

Die Schwerpunkte bei der Planung von flexiblen Fertigungssystemen lagen in den Anfängen der Automatisierung zunächst bei der Festlegung des Maschinenparks sowie der Auswahl der Transportmedien. Der nächste Schritt zur Bewältigung von Flexibilitätsengpässen war die Bemühung um die Maschinenperipherie, d.h. um die Gestaltung des maschinennahen Werkstück- und Werkzeugflusses. Erst allmählich wird das Ausmaß der organisatorischen und wirtschaftlichen Verbesserungen durch die Schlüsselfunktion Informationssystem im CAM-Bereich (Computer Aided Manufacturing) zur Steuerung der Bearbeitungsabläufe erkannt und es werden Anstrengungen zum Aufbau solcher Systeme unternommen.

Von der Vielzahl der Komponenten - wie Zuführung, Werkstück- und Werkzeugtransport, Steuerung, Arbeitseinrichtungen, Gestelle sowie Schutzeinrichtungen -, die beim Aufbau und beim Betrieb einer Montagezelle entscheidend sind, sollen deshalb besonders die Steuerungsaspekte und hier wiederum speziell der den elementaren Steuerungen (NC, RC, SPS) überlagerte Rechner, in der vorliegenden Form Zellenrechner genannt, betrachtet werden.

Um eine hohe Produktivität zu erreichen, ist eine bestmögliche Abstimmung zwischen den Montage-, den Handhabungsvorgängen und den Abläufen in der Peripherie erforderlich. Aus diesem Grund werden die einzelnen RC-Steuerungen bzw. NC-Steuerungen für Achsgeräte und die speicherprogrammierbaren Steuerungen des Transportsystems und der Peripherie über einen Zellenrechner gekoppelt. Dieser steuert, überwacht und koordiniert die Vielzahl der Einzelvorgänge und nimmt somit die Leitfunktion innerhalb der Zelle wahr.

Die Entwicklung eines "Standards" durch die Pionierrolle der IBM-PC's, die Verbesserung der Grundausstattung an Speichereinheiten und Schnittstellen sowie eine breite

Palette an Erweiterungsmöglichkeiten in Form von Steckkarten und der Integration von Echtzeitbetriebssystemen lassen den PC als prädestiniert für den Einsatz als Zellenrechner erscheinen.

Die anzutreffende Wandlung im Montagebereich liegt nicht in den Vorgängen innerhalb der Montageausführung, sie verbirgt sich vielmehr hinter dem Begriff der 'Flexibilität' und auch der 'Automation' innerhalb der Fertigungssteuerung. 'Automation' beinhaltet als Kernaussage zunächst das selbständige Ablaufen von Vorgängen ohne Eingriffe des Bedieners. Die Teilsysteme müssen also im laufenden Betrieb zur Erfüllung ihrer Aufgaben, d.h. zum Durchführen der Transporte (Zu- und Abführen von Material und Hilfsstoffen) und der Vorgänge wie Handhaben, Montieren und Prüfen von Teilen die nötigen Informationen erhalten.

Die Automation auf dem Gebiet der Teilefertigung wurde durch standardisierte formgebende Bearbeitungsvorgänge erleichtert. Erst nach und nach wird die Notwendigkeit erkannt, Beschreibungsformen und Standardisierungsansätze der komplexen Montagevorgänge zu finden, denn ohne diese Hilfsmittel kann eine Automatisierung jeweils nur problemgebunden durchgeführt werden /12, 20/. Die Flexibilität der Montagezelle beruht aber gerade auf der vielseitigen Verwendbarkeit ihrer Einzelkomponenten sowie deren Zusammenwirken. Die Berücksichtigung dieser Aspekte muß das Ziel neuer Konzepte innerhalb der Steuerungsproblematik einer FMZ sein.

2.2.2 Zellensteuerung mit neuem Profil

Der Rechnereinsatz innerhalb der Montageausführung hat in Zukunft sicherlich weitere Erfolgschancen, geht es doch gerade hier um komplexe Situationen und Reaktionen /7/. Die Aufgaben des klassischen Problemfeldes Werkstattsteuerung werden durch die Forderungen der Flexibilität und kleiner Losgrößen zahlreicher und zugleich vielfältiger. So muß auch die Frage nach Aufteilung bzw. Zerlegung von Aufgabenfeldern und nach Aufgliederung in hierarchische Steuerungssysteme gestellt werden. Viele bisher nur übergeordnet angesiedelte Aufgaben müssen daher näher an das Prozeßgeschehen, d.h. auf die Zellenebene verlagert werden. So ist für die Lösung der Problematik in der Gestaltung der Software im Zellenrechner die Umsetzung folgender Punkte entscheidend:

- Generierung aktueller Planungsvorgaben
- Vermeiden einer Planignoranz
- Vorgabe unterschiedlicher strategischer Ziele
- Gewährleistung eines automatischen Ablaufes

Diese vier Zielformulierungen sollen in Zusammenhang mit dem Zellenumfeld erläutert werden.

Generierung aktueller Planungsvorgaben:

Zum Steuern ist immer ein Plan, das Soll, erforderlich. Die Methodik, die Planung ist Aufgabe vom PPS bzw. in geringem Maße von der Werkstattsteuerung, ist aber nicht auf schnelle Reaktionsplanung ausgerichtet, wie sie zum sofortigen Reagieren auf bestimmte Zellenereignisse erforderlich sein kann. Dies führt dazu, daß zukünftig mehrfach, in verschiedenen Ebenen, mit unterschiedlichen Genauigkeitsgraden geplant werden muß. Die Präzision kann sich hierbei bis zur Echtzeitreaktion erhöhen.

Vermeiden einer Planignoranz:

Die in der Praxis anzutreffende "Planignoranz" ist zu meiden, d.h. obwohl man nach hochentwickelten Methoden plant, wird einerseits die Abarbeitung einer Warteschlange dem Werker / Leitsystem überlassen und andererseits für Störungen kein Vorschlag über weitere mögliche (geschweige denn suboptimale) Reihenfolgen gegeben. Dies bedeutet: im Grunde war der Plan wertlos, die Ziele der Planung werden verfehlt, weil sich die Planungsergebnisse in zufälligen Reihenfolgen niederschlagen. Somit muß eine Zellensteuerung hochaktuelle Daten bearbeiten und entsprechend der Lage entscheiden, d.h. der gerade gültige Plan (Reihenfolge, Zuteilung) darf nur aufgrund von aktuellen - nicht vermuteten - Ereignissen bzw. nach Genehmigung durch die Bedienperson geändert werden.

Vorgabe unterschiedlicher strategischer Ziele:

Ein Plan soll immer einem bestimmten strategischen Ziel folgen. So ist es überraschend, daß PPS-Systeme auch konkrete Vorgaben für Zellen durchführen, obwohl PPS-Pläne sich nur für übergeordnete strategische Ziele eignen. Um Vorgaben für Stationen innerhalb von Zellen zu bestimmen, müssen jedoch häufig unterschiedliche Methoden - genau ausgerichtet auf die dort geforderten Ziele - angewendet und ihr Erreichen wirkungsvoll unterstützt werden. Diese Methoden müssen vorgebbar und wählbar sein, d.h. in verschiedenen Steuerungsebenen, so auch in einer intelligenten Zellensteuerung, muß mindestens ein mögliches Ziel angestrebt und bei mehreren zwischen ihnen gewichtet werden.

Gewährleistung eines automatischen Ablaufes:

Ein rechnergestütztes Steuerungssystem muß einen Ablauf im Normalfall auch ohne Bedienereingriff ermöglichen und dies vor allem nicht nur beim Montieren genau eines

Erzeugnisses, sondern auch bei einer oder mehreren variantenreichen Produktfamilien. 'Sonderfälle' sind je nach Art und Weise zu erkennen und nach bestimmten Richtlinien (Bedienermeldung, automatisches Anzeigen, Strategieänderung) zu behandeln.

2.2.3 Gestaltung von Software

Software ist zwar ein immaterielles Produkt, verbraucht sich nicht, unterliegt keinem Verschleiß und erfordert keine Ersatzteile, muß aber als 'flexibelste' Komponente in einem Fertigungssystem die Anpassung und Integration durch relativ einfache Änderungen ermöglichen. Hier ist die Produktionsinformatik durch Verbindung von Fach- und Softwarewissen gefordert, denn nur durch methodisches Vorgehen kann erreicht werden, daß die Informationen der Konstruktion (z.B. Anlagen-, Teilebeschreibung), der Fertigung (z.B. Fertigungsverfahren, Montagevorschriften, Qualitätssicherung) und der Informatik (z.B. Rechnersysteme, Programmiersprachen) in eine Softwarelösung eingebracht und dort verwertet werden.

Die Entwicklungen setzen momentan einerseits genau auf einer Problembeschreibung auf und es werden kaum Aspekte der Aufbereitung bestehender Software herangezogen (Unwissenheit bzw. Unkenntnis der Funktionalität und Strukturierung). Andererseits versucht man aufgrund dieses Sachverhaltes völlig konfigurierbare Zellensysteme zu entwickeln. Hierbei ergibt sich aber folgendes Dilemma: Es müssen alle Anforderungen des Erstbenutzers in hinreichendem Maße abgedeckt werden (was teilweise schon nicht leicht ist), weiterhin müssen potentielle Anforderungen (nicht exakt formulierbar) von zukünftigen Benutzern berücksichtigt werden und dazu noch Möglichkeiten offengehalten werden, um Veränderungen durch zukünftige Entwicklungen einzubringen.

Da der letzte Teil dieser Aufgabe streng genommen nicht zu lösen ist, sind diese Entwicklungen der konfigurierbaren Zellensysteme nicht tragbar und großteils zum Scheitern verurteilt /44, 109/. Neben Konfigurationsproblemen und mangelnder Transparenz sowie Erweiterungsmöglichkeiten trägt auch der Kostenfaktor zur fehlenden Akzeptanz bei /44, 56, 115/.

Man muß sich also auf Konstruktionsprinzipien, Entwurfskriterien und Realisierungsmethoden zurückziehen, die möglichst günstige Voraussetzungen dafür schaffen, daß die jeweiligen Anforderungen systematisch gelöst werden können und auf zukünftige Bedürfnisse durch Erweiterungsansätze (offene Systeme) reagiert werden kann.

2.3 Stand der Technik und Entwicklungstendenzen

Die gegenwärtige Welt der Anwendungssoftware ist gerade innerhalb des CAM Bereiches sehr heterogen. Im Bereich der zeitnahen Fertigungssteuerung fehlt innerhalb der Standards gegenwärtiger PPS Systeme jegliche funktionale Ausrichtung. Auf der fertigungsnahen Ebene dominieren die PC's, für die neben Softwaretools zur Erstellung von Anwendungen und bestimmten Kommunikationsdiensten zwischen Zellenrechnern und unterschiedlichen Steuerungen aber gerade werkstattorientierte Softwaresysteme benötigt werden /89/.

Vorhandene Systeme zeigen jedoch auf der Ebene der Steuerung von Zellen sehr spezialisierte Anwendungserscheinungen, so daß sich Hersteller nicht auf einheitliche Anwendungssoftware wie auf höherer Ebene, z.B. PPS System COPICS, beschränken können. Somit sind Architekturen gefordert, die durch Schlagworte wie portable Applikationen, Endgeräteunabhängigkeit, Datenzugriffsunabhängigkeit und Betriebssystemunabhängigkeit gekennzeichnet werden können, wodurch gerade die Bedeutung von Datenstrukturen bei der Bildung von Anwendungssystemen unterstrichen wird.

2.3.1 Aspekte von Verfahrensketten

Innerhalb von Verfahrensketten wird größtenteils versucht, Programmiersysteme weiterzuentwickeln, die auf Gegebenheiten der rechnergestützten Montageplanung aufbauen /23, 30, 62/, z.B. mit Hilfe von Hochsprachen zur Roboterprogrammierung. Weitere Ansätze liegen bei der Verarbeitung von Korrekturwerten direkt am Steuerungsgerät, z.B. Werkzeugkorrekturen und Nullpunktverschiebungen.

Ein Ziel - speziell für die Montage - ist daher die Entwicklung und Realisierung eines Konzeptes für die Generierung von Parametern in NC/RC Programmen, um nicht für jede Produkt/Baugruppen - Variante ein neues NC/RC Programm von einem Programmiersystem anzufordern. Der Aspekt der Generierung von Programmparametern vor Ort für Robot-Control und Numerical-Control Steuerungsgeräte bildet dann die Basis, um Abläufe in gewissen, vorgeplanten Grenzen variabel zu gestalten, z.B. Variation von Geometrie- und Positionsdaten.

Durch Ansätze zur Vereinheitlichung von Schnittstellen für die strukturierte Übergabe von Geometrie- und Technologiedaten können Entwicklungen auf Zellenebene stärker in den Vordergrund rücken und konkretisiert werden /19, 94/. Bisher müssen jeweils Ver-

einbarungen getroffen werden, damit eine Übertragung von Daten zwischen verschiedenen rechnergestützten Systemen mit Hilfe von Konvertierungsprozessen möglich wird, was ein weiteres Hemmnis zur Integration von Generierungsmechanismen in der Zellenebene darstellt.

Somit werden pro 'Montageaufgabe' in der Arbeitsvorbereitung aufbauend auf den Daten über die Produktionsmittel aus der Fertigungsplanung und den Produktdaten aus der Konstruktion die Steuerprogramme für die im Montagesystem befindlichen Montagemodule (Gerätesteuerungen) erstellt. Diese Daten werden dazu mit Wissen über die Montagevorgänge und Technologien ergänzt /88/ und repräsentieren genau einen festen Ablauf. Allen Entwicklungen der Programmiersprachen für Industrieroboter ist gemeinsam, daß immer mehr Aufgaben in die Robotersteuerung integriert werden, um so die Echtzeitfähigkeiten der IR-Steuerung zur Koordination aller Komponenten einer Zelle oder einer Station auszunützen /13, 37/. Die IR-Steuerung zur Koordination von 'verknüpften Aufgaben' heranzuziehen, die keinen Echtzeitcharakter aufweisen, kann aufgrund mangelnder Rechnerkapazitäten die Flexibilität des Systems enorm einschränken und zusätzliche Schwierigkeiten wegen der Sprachvielfalt aufwerfen.

So wird versucht, eine Vielzahl von bereits genannten Unzulänglichkeiten der Programmiersprachen durch teilweise aufwendige Programmiertechniken zu umgehen. Zusammenstellungen der Programmierverfahren mit ihren Anwendungsgebieten zeigen auf, daß On-Line-Verfahren nur dann sinnvoll sind, wenn sich Produkte oder Verfahren selten ändern oder wenn es sich um spezielle Montageverfahren handelt, deren Parameter erst erarbeitet werden müssen /60/. Werden dagegen häufig Stationskonfigurationen und Produkt geändert, bieten Off-Line-Verfahren den Vorteil, daß die Stillstandszeiten des IR-Roboters reduziert werden, aber haben auch den Nachteil, daß pro Änderung der Konfiguration oder pro Produktvariante ein eigenes Programm erstellt werden muß.

Insgesamt bleibt als grundsätzliches Problem die fehlende Standardisierung der Robotersteuerungen, die die Schaffung offener Systeme erschwert. Weiterhin sind, so auch bei IRDATA, keine Sprachmittel für den Informationsfluß von der Robotersteuerung zum 'Leitrechner' vorgesehen. Durch fehlende Integration von BDE und CAQ Diensten ist der Aufbau prozeßnaher Regelkreise behindert.

Die Einbindung von Sensorinformationen, die viele Roboteranwendungen erst ermöglicht, ist geräteabhängig und kann deshalb schwer in übergeordneten informationsaufbereitenden Prozessen berücksichtigt werden. Die Vermischung von Programmstruktur, technologischen (z.B. kinematische Parameter) und geometrischen Daten schafft eine

starke Abhängigkeit von Produkt und Zellenkonfiguration. Dies wirkt einschränkend auf die Produkt- und Erweiterungsflexibilität. Außerdem können Planungsmethoden zur Programmerstellung die Redundanzen in den häufig ähnlichen Vorgängen nicht nutzen, da jede Änderung der Geometrie einen neuen Planungslauf erfordert /17/.

Wegen der Vielzahl der Aufgaben bezüglich der Steuerung, Koordination und Überwachung der Prozeßebene mit ihren unterschiedlichen Ausprägungen wird immer mehr von einer zentralen Steuerungsstruktur mit nur einem Leitstand abgewichen. Mehr und mehr faßt man Gruppen von Maschinen zusammen und setzt einen überlagerten Rechner darüber, der aktuell in der Verfahrenskette meist DNC und / oder BDE (organisatorischer Aspekt) Aufgaben übernimmt /31/. Auf unterschiedlicher Ziel-Hardware wird vielerorts so je Maschinenzusammensetzung und für ein ausgewähltes Aufgabenspektrum eine konkrete Applikation geschaffen, die aufgrund fehlenden systematischen Vorgehens nur 'aufgabenbezogen' konzipiert wird und somit kaum eine Übertragung von Aspekten auf ein anderes System zuläßt. Die herkömmliche Schnittstelle zwischen Planungsebene und Maschinenebene stellt das NC / RC Programm dar. Diese Schnittstelle kann aber grundsätzlich Kriterien flexibler Zellenstrukturen wie

- Unabhängigkeit der Steuerungsstruktur von Aufgaben- und Zellenausprägung
- Berücksichtigung von Zellenzustandsdaten (online)
- Strukturierte Informationsrückgewinnung
- Unterschiedliche Methoden der Gesamtintegration
  (z.B. autarker Betrieb, Anschluß an Leitsysteme)
- Aufwandsreduzierung für Realisierung und Betrieb

nicht gerecht werden.

Weiterentwicklungen betreffen einerseits - wie oben erwähnt- die herkömmlichen Technologien, andererseits gibt es aber auch Bestrebungen, neue Konzepte zu finden, um die grundsätzlichen Probleme in diesem Bereich zu vereinfachen. Die implizite oder aufgabenorientierte Programmierung stellt dabei eine Weiterentwicklung der bisherigen Steuerungskonzepte dar /37/. Neben der Modellierung des Umweltmodells, der Kollisionsvermeidung, der Bahnplanung müssen Greifplanungen und die Integration von Sensorinformationen vollzogen werden. Es bleibt aber wiederum die fehlende Flexibilität der Steuerung auf der operativen Ebene (Reaktion im Rahmen von Zustandsdaten), die durch großen Aufwand bei der Planung kompensiert werden soll.

Bei Berücksichtigung von Zustandsdaten sind so z.B. im Umweltmodell der augenblickliche Montagefortschritt und die Ortsveränderungen der Objekte (Bereitstellungsort in einem Magazin) zu protokollieren. Wird aber bei der Planung auf sich mit der Zeit veränderbare Daten zurückgegriffen, so müssen Teile des Planungsprozesses online statt-

finden. Allerdings stellt sich bei der Wandlung des Roboterprogrammes von einem statischen zu einem dynamischen Gebilde die Frage, ob die rechenaufwendigen Planungsprozesse (Kollisionsvermeidung, Bahnplanung), deren Realisierung bereits offline Probleme bereitet, online in vertretbarer Zeit bewerkstelligt werden können.

2.3.2 Hierarchische Steuerungskonzepte

Allen obigen Ansätzen ist gemeinsam, daß problemangepaßte Lösungen entstehen, deren Flexibilität begrenzt ist. Aus diesem Grunde entstanden Ansätze, die versuchen, die Planungs- und Ausführungsaufgaben neu zu strukturieren und damit Modelle aufzubauen, die ohne 'Reibungsverluste' die Informationen optimal nutzen können. Grundlegender Gedanke ist jeweils ein Schichtenarchitekturmodell, das die Planungs- und Ausführungskompetenzen jeder Ebene festlegt und dadurch Prozesse definieren kann, die systematisch realisiert werden können (Referenz- und Realisierungsaspekt).

Neben Zörntlein /117/, dessen Modell ein Konzept zur Unterteilung von Steuerungsfunktionen für ein flexibles Fertigungssystem darstellt, kann auf ein Schichtenarchitekturmodell von Scholz /95/, das sich auf ein Produktionssystem bezieht, verwiesen werden. Zörntlein hält zwar ebenfalls an festen Programmstrukturen in Verbindung mit den umweltbeschreibenden Daten fest, schafft aber durch die Strukturierung der Steuerungsaufgaben die Möglichkeit, an den Schnittstellen Planungskompetenzen einzufügen.

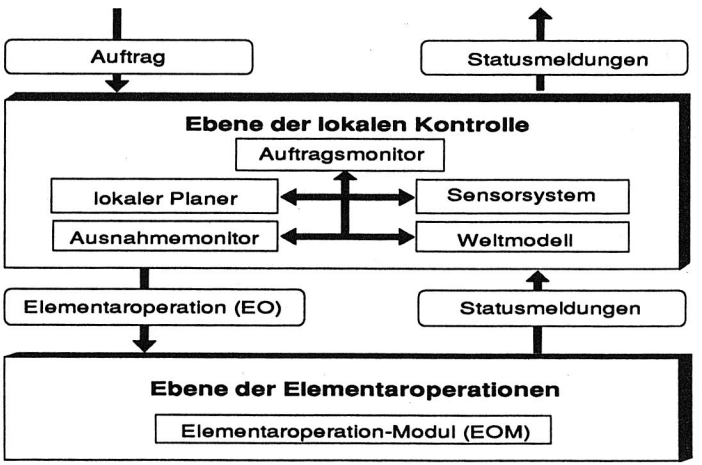


Bild 3: Modell einer Agentensteuerung

Beim Konzept nach /47/ für eine Zellensteuerung liegt der Schwerpunkt auf der Fehlertoleranz, die als unmittelbare Berücksichtigung der Abweichungen des Weltmodells von der Wirklichkeit beschrieben wird. Es wird also ein Steuerungskonzept entwickelt, das reaktive und plangesteuerte Verhaltensmuster verbindet (Bild 3).

Hier steht also ein System mit verteilter Intelligenz im Vordergrund, das zur Bewältigung der obigen genannten grundsätzlichen Probleme dient. Dieser Ansatz setzt eine vollständige Rechnerintegration aller steuernden und planerischen Komponenten voraus und stellt somit eher einen Ansatz in Richtung grundlegender Forschungsschwerpunkte dar. Ähnlich der aufgabenorientierten Programmierung, falls diese als Online-Planungssystem verstanden wird, erscheint eine Realisierung unter wirtschaftlichen Gesichtspunkten vorerst nicht möglich und deshalb können hieraus kaum Veränderungen für die heutige Montagewelt abgeleitet werden.

So scheinen sich Realitätsnähe und Intelligenz auf der steuerungsnahen Ebene momentan gegenseitig auszuschließen, denn verbindende Ansätze, die tragfähige Kompromisse ermöglichen, fehlen (vergl. Bild 4).

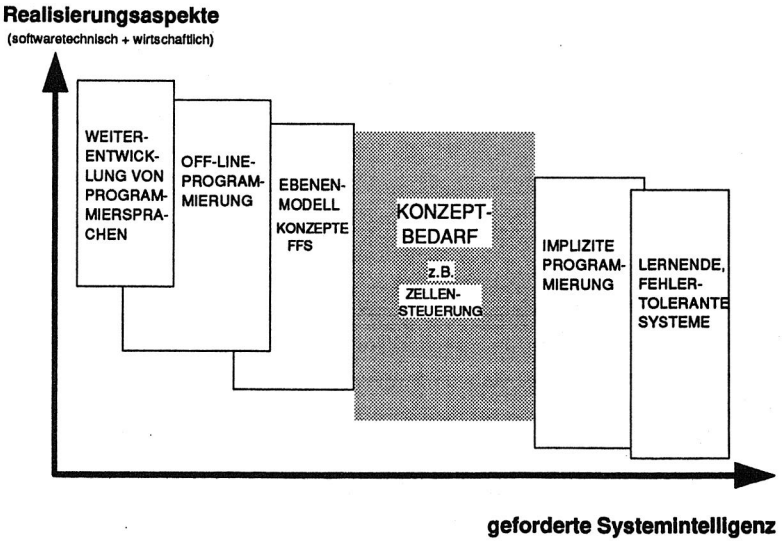


Bild 4: Entwicklungstendenzen in bezug auf prozeßnahe Montageaufgaben

2.3.3 Normierungs- und Standardisierungsbestrebungen

Zur Übertragung von produktionstechnischen Daten in der Teilefertigung steht CLDATA (Cutter Location Data - DIN 66215/ISO 8632) zur Verfügung, womit Daten maschinenneutral weitergegeben werden können. Zur Übertragung von RC-Daten (Robot Control) wurde IRDATA (Industrial Robot Data) vereinbart, das sich an CLDATA orientiert und identische Eigenschaften für den RC-Bereich besitzt. Dabei wird von einer einseitigen Übertragung vom CAP (Arbeitsvorbereitung) zum CAM (Produktionsausführung) ausgegangen; eine Rückkopplung ist somit nicht möglich.

Um jedoch auch produktorientierte Daten übermitteln zu können, werden von nationalen und internationalen Organisationen Anstrengungen zur Entwicklung von Standards unternommen. Zu nennen sind hierbei PDDI (Product Definition Data Interface), PDES (Product Data Exchange Spezification) und für den Elektroniksektor das Datenaustauschformat EDIF (Electronic Design Interchange Format). Produktorientierte Daten beinhalten einerseits geometrische, andererseits aber auch technologische und organisatorische Informationen, die Angaben über Werkstoff, Toleranzen, Oberflächengüte, Normteile und Stücklisten sein können /19, 95/.

Beim Aufbau von Verfahrensketten existieren in der Normungswelt zwei Referenzmodelle: das Produktmodell (produktdefinierende Daten) und das Betriebsmittelmodell (fertigungs-, montage- und prüfrelevante Informationen über Betriebsmittel). Hinsichtlich des dynamischen Verhaltens der Fertigungssteuerung und der Auftragsabwicklung in der Werkstatt können diese Modelle als Basis betrachtet werden. Zur Übertragung produktdefinierender Daten sind Ansätze zu Elementenfamilienmodellen, die sich durch eine begrenzte Anzahl von Parametern beschreiben lassen, vorhanden /19/, genauso wie die Beschreibung der Produktgestalt durch Geometrie und Topologie /3, 41/.

Dies sind Voraussetzungen dafür, daß auch in einem Zellenrechner (Informationssystem vor Ort) eine Strukturierung auf Basis dieser Gesichtspunkte vorgesehen werden kann, z.B. die Abspeicherung von funktionalen Zusammenhängen zwischen Produktkomponenten (Strukturgraph) und Regeln zur Erzeugung von Stücklisten aus der Produktstruktur und Erzeugnisgliederung. Für eine Anwendung auf Zellenebene können dann die zu übertragenden Schnittstelleninhalte ermittelt und logische Darstellungsschemata spezifiziert werden, um hieraus auf die jeweils zur Realisierung notwendigen Daten und Datenstrukturen zu schließen.

Die jedoch noch eingeschränkten Einsatzmöglichkeiten von den bisher existierenden Schnittstellenstandards (meist auf bestimmten Problembereich bezogen) für eine Integration im CIM-Sinne (z.B. IGES (Initial Graphics Exchange Specification), SET (Standard d'Echange et de Transfer), PDES (Product Definition Exchange Specification)) haben das Ziel des weltweiten Normungsvorhabens STEP (Standard for the Exchange of Product Model Data) spezifiziert:

> Die Entwicklung einer internationalen Norm, die ein Produktmodellschema und Übertragungsformat definiert, das alle im Produktlebenszyklus entstehenden Informationen beinhaltet.

Hierbei gehören zum Produktlebenszyklus auch Informationsmengen aus dem Produktionsbereich. Es existieren somit neben Modellen zur topologisch/geometrischen Objektbeschreibung auch Modelle für Arbeitsplanungsdaten und Produktstrukturdaten /42/. Zielsetzung normungsbegleitender Forschungsvorhaben bleibt neben einer Sicherstellung eines standardisierten und genormten Datenaustausches zwischen DV-Applikationen auch die Gewährleistung, daß unterlagerte Systeme die aufbereiteten Daten verwerten können.

Als Basis existiert ein Referenzmodell, das aus dem Produktmodell, dem Betriebsmittelmodell sowie den Anwendungsprogrammen und jeweiligen Schnittstellen besteht. Die Anwendungsprogramme bearbeiten a priori Produktdaten und greifen hierbei auf Betriebsmitteldaten zu. Das Betriebsmittelmodell kann sämtliche Produktionsmittel eines Unternehmens, die zur Herstellung der Produkte benötigt werden, beschreiben oder auch nur Teile hieraus, die dann z.B. das Potential einer Montagezelle (Planungsdaten) darstellen /23/.

Das Produktmodell ist die Dokumentation aller relevanten internen und externen Daten, Vorgänge, Informationen und Ergebnisse, die für einen Auftrag von der Angebotserstellung über die Fertigung und Auslieferung bis zum Ablauf der Produktverantwortung anfallen. Das Produktmodell wird für eine Verfahrenskette originär und in erster Linie von CAD-Systemen gefüllt, um Geometrie und Technologie für die weiteren Planungen bereitzustellen. Auf dieses Produktmodell greifen die einzelnen Systembausteine einer Verfahrenskette zu, um einerseits ihren Informationsbedarf zu befriedigen und andererseits auch erzeugte Informationen wieder einzuspeichern (Bild 5).

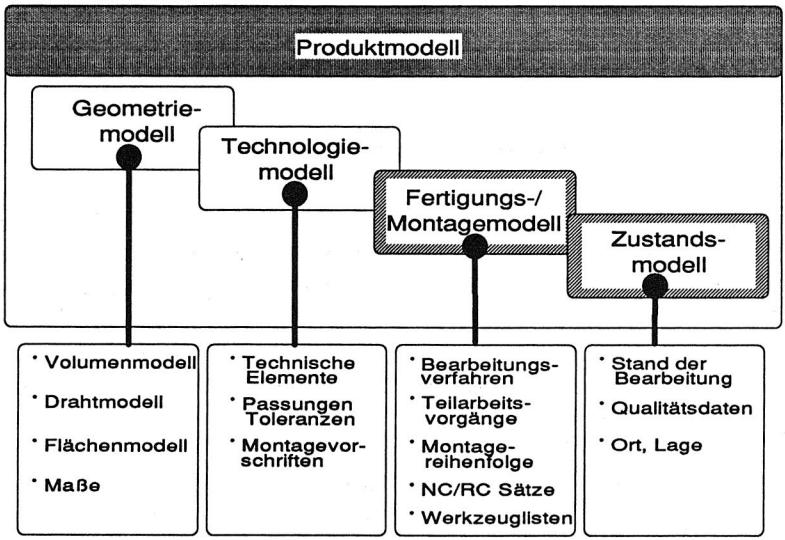


Bild 5: Produktmodell

Die Werkstattsteuerung und somit auch die Zellensteuerung als Glied in der Kette haben die Aufgabe, basierend auf den Planungsdaten, die im sogenannten Produktmodell bzw. Betriebsmittelmodell gespeichert sind, den Fertigungsprozeß mit seinen Funktionen (z.B. Montieren, Prüfen) zu steuern und in der Prozeßausführung gewonnene Daten in das jeweilige Modell einfließen zu lassen.

Die Übertragung berührt sowohl technische als auch organisatorische Daten. Eine mögliche Gliederung kann durch die Begriffe DNC (Direct Numerical Control) und BDE gegeben werden. Zukünftig sollte jedoch nicht nur Wert auf die bisherigen Schwerpunkte bei diesen bekannten Begriffen gelegt werden, d.h. die Vorgabewerte sollen nicht nur auf technische Daten wie z. B. bei IRDATA ausgerichtet sein und die Rückmeldungen nicht nur auf die Erfassung organisatorischer Daten eingehen.

So spielen nun auch vorgangsbezogene Daten, produktbeschreibende Daten sowie organisatorische Daten eine Rolle. Durch eine funktionale Einteilung der Daten können dann die Zusammenhänge und Wechselwirkung verdeutlicht werden (z.B. Einsatzplandaten in bezug auf die Betriebsmittel). Diese zunächst vorwiegend in der Arbeitsvorbereitung entstandenen Daten (typspezifischen Angaben) werden an Leit- und Zellenebene weiter-

gegeben und dann mit exemplarspezifischen bzw. zustandsspezifischen Werten verrechnet, um so auch als Angaben innerhalb der Feinterminierung bzw. Betriebsmittelprüfung vor Ausführung von Aktivitäten verwertbar zu sein.

2.3.4 Qualitätssicherung der Software

Flexible Fertigungszellen (FFZ) und Flexible Montagezellen (FMZ) stellen moderne Automatisierungskonzepte dar /85/. Es existiert jedoch für beinahe jede Automatisierungslösung eine individuelle Softwarelösung. Neben den fertigungstechnischen Kosten (Maschinen etc.) tragen die Kosten für die Softwareerstellung wesentlich zu hohen Gesamtinvestitionskosten /103/ bei. Nicht standardisierte Vorgehensweisen bei der Softwareentwicklung behindern Rationalisierungseffekte wie Modularität, Erweiterbarkeit, Wiederverwendbarkeit, Fehlerfreiheit und Zuverlässigkeit /63, 92/.

Die vorzufindende starre, an eine konkrete Konstellation angepaßte Software ist meist aufgrund unzulänglicher Analysen und Entwürfe aus unterschiedlichen Sichtweisen entstanden und dazu aufgrund fehlender Entwicklungszeiten nur unzulänglich getestet und weist entsprechend viele Fehler auf. Weiterhin treten bestimmte Fehlerfälle nur bei bestimmten Systemkonfigurationen auf und werden für ein bestimmtes Projekt nicht beachtet bzw. können nicht nachvollzogen werden.

Diese nicht erprobten Softwarelösungen mit ihren Softwareschwächen führen gerade beim Anlauf eines rechnergeführten Fertigungssystems zu "teueren" Verzögerungen. Treten Softwarefehler erst später auf (beispielsweise während des Systembetriebs), dann wirken sich auftretende Schäden am Fertigungssystem bzw. Verzögerungen im Fertigungsergebnis besonders negativ auf die Systemnutzung aus.

Der industrielle Einsatz von Zellenrechnersoftware ist in der Regel gekennzeichnet durch hohe Entwicklungs- und Pflegekosten und lange Entwicklungszeiten. Ein Vergleich mit der Hardwareentwicklung zeigt, daß dort in den letzten Jahren Produktivität und Qualität durch intensiven Einsatz von Konzepten der Wiederverwendbarkeit erheblich gesteigert wurden. Die Softwareentwicklung stellt aber in bezug auf die Qualitätssicherung einen Engpaß dar und dies besonders beim Einsatz im Produktionsbereich (Steuerung, Überwachung).

Somit bietet gerade die Forderung nach Konzeptionen zur effektiven Softwaregestaltung eine Motivation, um Wege zur schnellen Erstellung von Software im Automatisierungs-

bereich zu finden, die gleichzeitig eine positive Wirkung auf den Gesamtprozeβ haben. Hier wäre z.B. eine kürzere Anlaufzeit eines Systems oder eine geringere Ausfallszeit zu nennen.

Die Qualitätssicherung in der Automatisierung beschäftigt sich mit Maβnahmen am Produkt, am Fertigungsverfahren und an der eingesetzten Software /85/. Für die Qualitätssicherung (QS) der Software lassen sich allgemein nur Ansatzpunkte erkennen /4/, die die Phasen des Software-Lebenszyklus, die Phasenübergänge im Software-Lebenszyklus, die Versionshaltung und das Softwareprojektmanagement betreffen.

Werkzeuge, die speziell der Software-Qualitätssicherung in der Fertigungsautomatisierung dienen, werden heute auf dem Markt erst in geringem Umfang angeboten /46/. So liegen Schwachpunkte einerseits in der Umsetzung der fachlichen Spezifikation in eine ablauffähige Form (z.B. wie können Prüfregeln für Daten in einem konkreten Applikationsgebiet überhaupt aufgestellt bzw. automatisch erzeugt werden). Andererseits kommt hinzu, daβ gerade in der Betriebsphase auftretende Erkenntnisse meist nicht 'kontrolliert' und 'dokumentiert' in das Softwaresystem integriert werden, da Vorgehensweisen und Kontrollmechanismen vor Ort nicht mehr bekannt bzw. nicht mehr eingesetzt werden. Es ist jedoch notwendig, der Softwareentwicklung ähnliche Qualitätssicherungswerkzeuge zur Verfügung zu stellen, wie sie in der Fertigungstechnik seit langem Verwendung finden.

Mit der Planung allgemeiner Aspekte wiederverwendbarer Software und derer Anforderungen beschäftigen sich aufgrund der Komplexität sehr viele wissenschaftliche Untersuchungen /63, 82/. In der Automatisierungstechnik ist aber das Problem, daβ bei Einführung der Informationsverarbeitung gleichzeitig fachbezogenes Sachwissen und mathematisch - algorithmische Durchdringung und Beschreibung der Vorgänge vorgenommen werden muβ. Durch die Konzentration der Anwendungs- und Systemexperten auf eine jeweilige Lösungsmöglichkeit entstehen monolithische Softwaresysteme, in denen auch das Wissen über den Ablauf der Vorgänge begraben wird.

In jeweiligen Anwendungsbereichen der Fertigungsautomatisierung (CAM) ist somit neben der Steigerung der fertigungstechnischen Qualität eine grundlegende Verbesserung der softwaretechnischen Qualität unter Einbeziehung von Hardware und Systemsoftware unabhängiger logischer Strukturen - vergleiche hierzu auch /40/ - dringend erforderlich .

2.4 Zielsetzung und Vorgehensweise

In vielen Bereichen macht die Abkehr von der Inflexibilität der industriellen Produktion und der Trend hin zur entspezialisierten Fertigung und hierarchischen Steuerungsstrukturen in der flexiblen, produkt- und variantenreichen Mittel- und Kleinserienfertigung neue Steuerungskonzepte zur Erfüllung der Aufgaben notwendig. Durch den Einsatz rechnergeführter Produktionssysteme tritt die Frage nach Erfüllung der Termin-, Kosten- und Qualitätsforderung auch beim Produkt 'effektive' und 'gute' Steuerungssoftware in den Vordergrund.

Ziel der Anstrengungen sollte somit sein, durch gezielten Rechnereinsatz zur Erreichung der jeweils geforderten Flexibilität beizutragen. Auf Zellenrechnerebene steht jeweils die Klasse von dispositiven prozeβnahen Anwendungsaufgaben als Problem zur Lösung an. Neben Methoden zur Softwareerstellung sind vor allem Architekturkonzepte notwendig, die den Benutzer bei der Planung, beim Aufbau und Betrieb von Steuerungssystemen wirkungsvoll unterstützen. Als Voraussetzung dafür, daβ eine weniger fehleranfällige Software entsteht, müssen Aspekte der Abstraktion und der Wiederverwendbarkeit in den Vordergrund gestellt werden.

Wie aus den vorhergegangenen Betrachtungen ersichtlich, sind viele Arbeiten auf dem Gebiet der Planung (z.B. Programmiersysteme) und der direkten Gerätesteuerungsebene angesiedelt. Für die komplette Durchgängigkeit jedoch sind Zellenrechnerstrukturen notwendig, die in einen Verbund integriert werden können und durch geeignete Funktionen und Mechanismen geforderte Flexibilitätsaspekte erfüllen.

Deshalb sind gerade Workstations oder Personalcomputer mit UNIX Betriebssystemen bzw. mit UNIX Derivaten (z.B. XENIX) für den Einsatz als Zellenrechner interessant, um eine Verbindung sowohl mit übergeordneten Systemen als auch mit Gerätesteuerungen bzw. Rechensystemen, die die Engpässe im Zeitverhalten beim Einsatz zur direkten Gerätesteuerung aufheben, herzustellen und gleichzeitig eine gewisse Intelligenz aufweisen.

Aufgabe ist deshalb die Entwicklung einer geräteunabhängigen Steuerungslogistik - zunächst im Hinblick auf ein eingegrenztes Anwendungsfeld (hier der Steuerungsaspekt in rechnergeführten Montagezellen), die unterschiedliche Sichtweisen integriert und unterschiedliche Systemausprägungen (z.B. spezielle Anforderungen an den Ablauf) berücksichtigt.

Ausgehend von Funktionsgruppen im CAM Bereich und von Analysen typischer Anforderungen in bezug auf Flexibilität in Montageanlagen wird aufgezeigt, wie effektive Unterstützung bei einer Steuerungskonzeption (Vorgehensweise, Strukturierungs- und Wiederverwendbarkeitsaspekte) aussehen kann. Hierbei wird ein Zellenrechner mit einem Multitasking Betriebssystem betrachtet, der autark seine Vorgaben ausführt und auf aktuelle Daten der Betriebsmittel, des Auftragspools und vorgegebener Planungskriterien zurückgreift. Die oberste Zielsetzung heißt, eine 'offene' Montagezellenarchitektur mit internem Regelkreis zu entwickeln (Bild 6), die als Grundlage für die Aufbereitung einer jeweiligen Steuerungslogistik dient und folgende Forderungen erfüllt:

- Aufbau- und Ablauforganisationsflexibilität in der Zelle
- Anpassungsfähigkeit an Veränderungen des Umfeldes und der Montageprozesse, d.h. rechtzeitige Verfügbarkeit der richtigen Informationen am richtigen Ort
- Integration von Auftragsaufbereitung für Produktfamilien
- Abarbeitung von Montageaufträgen in bezug auf aktuelle Zustandsinformationen
- optimale Verwendung der Informationstechnologien (Datenbanken, Kommunikation, Softwaretechnologie)
- Verwendung von existierenden Programmen bzw. Funktionen
- Integration von Echtzeitsteuerungen und Anbindung von Maschinensteuerungen unterschiedlicher Hersteller
- Berücksichtigung von Aspekten verteilter Systeme und Multitasking-Systemen

Da bei Systemen der Automatisierungstechnik der Softwareanteil heute ein bestimmender Faktor für Funktionalität, Qualität und Kosten ist, muß eine weitere Zielsetzung sein, einen Weg zu beschreiten, der dem Entwickler Hilfen für einen Softwareentwurf innerhalb des Aufgabenspektrums eines Anwendungsgebietes gibt und dort Aspekte der Wiederverwendbarkeit von Funktionsblöcken aufzeigt.

Das Anliegen muß also in dieser Situation sein, die besagten monolithischen Softwaresysteme in ein Netz kleiner überschaubarer Einheiten aufzubrechen, die bei sich ändernden Anforderungen flexibel zusammengeknüpft werden können.

Dies erfordert, daß eine Fachgebietsanalyse /70/, auch als domain analysis bezeichnet, in diesem Anwendungsgebiet erfolgt, um Grundstrukturen, die herausgezogen werden können, zu erarbeiten. So ist zunächst anwendungsübergreifendes Wissen bereitzustellen, das jeweils erst mit Systemwissen einer Anwendung kombiniert eine konkrete Ausprägung ergibt. D.h. es wird hier kein einzelnes passendes algorithmisches Modell für eine gegebene Anwendung gesucht bzw. gefunden, sondern ein gemeinsamer Rahmen geschaffen.

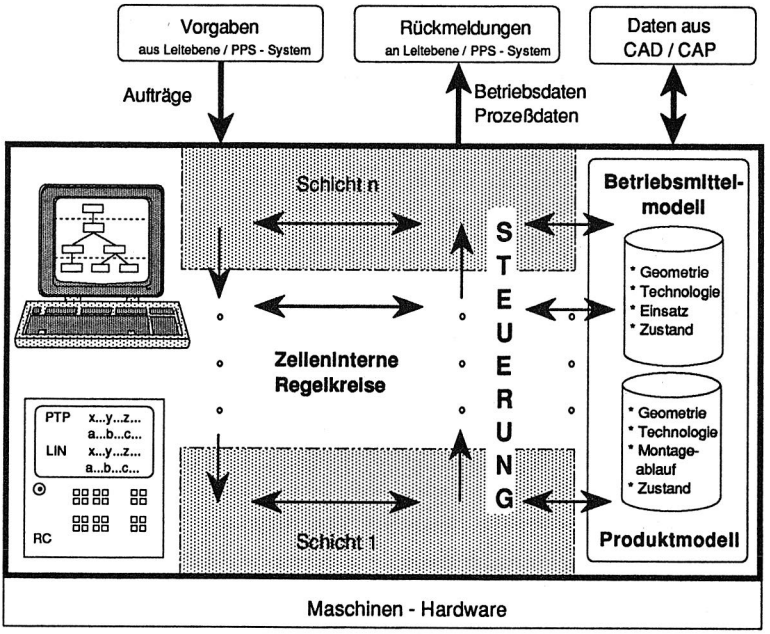


Bild 6: Konzeptrahmen

Anhand systematischer Untersuchungen werden Strukturen in bisherigen Arbeitsplänen und Zellensteuerungen dargestellt und Anforderungen aufgrund ausgewählter Aufgabenspektren abgeleitet. Diese Basis dient neben den Modellierungsgrundlagen zur Bildung von Architekturprinzipien und Informationsstrukturen für rechnergeführte Montagezellen. Zur Verwirklichung der genannten Zielsetzungen wird auf den Einsatz abstrakter Datentypen, relationaler Datenmodelle und Kommunikationsroutinen zurückgegriffen.

Eine detaillierte Behandlung der logischen Abstraktionsebenen und ihrer Daten mit formalen Beschreibungsmechanismen führt zu Funktionsmodulen und zur Darstellung wichtiger Parameter. Unter Zuhilfenahme dieser Prinzipien und Verfahren werden Steuerungssysteme für eine autarke Zelle zur Tastaturmontage und für eine in ein Fertigungssystem integrierte Pilotanlage entwickelt, realisiert und die Funktionsfähigkeit und Eignung der Software nachgewiesen.

3 Aufgabenorientierte Analyse

Eines der wichtigsten Kennzeichen einer Integrationsidee von Steuerungssystemen in ein CIM-Konzept ist der durchgängige Informationsfluß zwischen einzelnen produktionstechnischen Instanzen. Diese bereichsübergreifende Nutzung von Informationen sollte auch für Zellensteuerungen als Komponente eines rechnerunterstützten Gesamtkonzeptes im CAM-Bereich relevant sein (Bild 7) /100/.

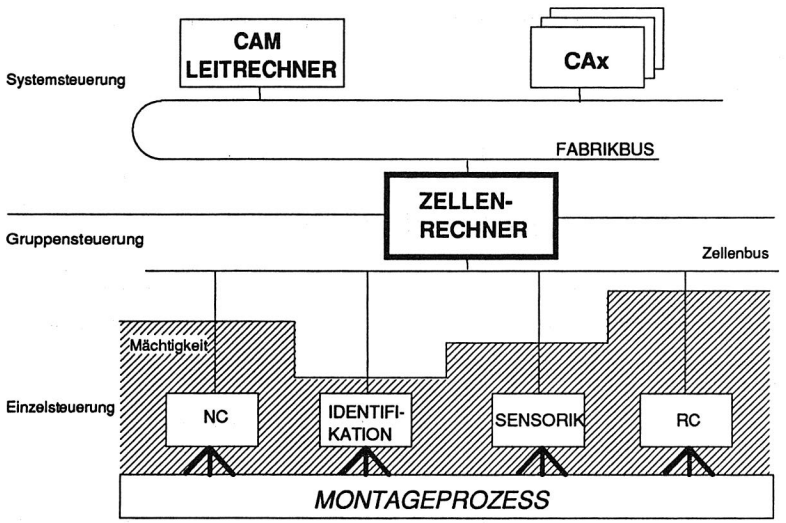


Bild 7: Zugrundegelegte CAM Steuerungsstruktur

Für rechnergestützte Montagezellen sind somit Anforderungen an die externe und interne Auftragsabwicklung zu stellen, um sowohl eine Integration in einen Gesamtverbund als auch intelligente Zellensteuerungssysteme an sich realisieren zu können. Erst eine Verbindung beider Aspekte trägt zur Verminderung sogenannter Zielkonflikte bei.

Viele Aufgaben eines Zellenrechners in der Montage sind sicherlich mit denen der Teilefertigung vergleichbar, so z.B. die Versorgung von Steuerungsgeräten mit Daten (Steuerprogramme), die Steuerung der Prozeßtechnologie (Verfahrensschritte mit individuellen Meß-, Einfluß- und Störgrößen), die Steuerung der Mensch - Maschine Kommunikation zur Anzeige und Diagnose von Systemzuständen sowie zur Entgegennahme von Operationsbefehlen durch Bedienpersonal. Zu der Aufgabengruppe, die sich durch die

Unterschiede im Arbeitsprozeß ergeben, gehören in diesem Rahmen vor allem die Teileverwaltung und -disposition sowie Koordinations-, Synchronisations- und Überwachungsaspekte (z.B. in bezug auf bereitgestelltes Material für unterschiedliche Aufträge und Varianten). Dies wiederum zieht andere Verwaltungs- und Zuordnungsmechanismen nach sich.

Die eingesetzte Steuerungssoftware zur Bearbeitung der Vorgaben (Aufträge, Arbeitspläne, Roboterprogramme, etc.) muß somit unterschiedlichsten Anforderungen genügen. Im folgenden wird aufgezeigt, welche Gestaltungskriterien zu beachten und zu erfüllen sind, um das Ziel 'intelligente' Softwaresysteme für Zellenrechner in der Montage zu erreichen.

3.1 Flexibilitätsaspekte in einer Zellensteuerung

3.1.1 Konkretisierung des Begriffs 'Flexibilität'

Flexibilität beschreibt allgemein die Fähigkeit, sich wechselnden Situationen anzupassen. Bezogen auf Produktionssysteme findet man die Unterteilung in Mengen-, Varianten- und Nachfolgeflexibilität, bezogen auf Fertigungssysteme unterscheidet man die Maschinenflexibilität (auch Technologieflexibilität genannt) und die Steuerungsflexibilität mit den Einflußparametern Produkt, Auftrag, Verfahren, Ausführung, Routing sowie Erweiterung.

Der zu beobachtende Trend zur Kleinserienfertigung bewirkt, daß der Begriff Flexibilität in zunehmenden Maße in Verbindung mit der Automatisierung von Montagevorgängen benutzt wird. Allerdings zeigt das Schrifttum /32, 65/ die Bedeutungsvielfalt und Unschärfe des Begriffs Flexibilität. Eines haben jedoch alle Deutungen gemeinsam:
Ein flexibles Montagesystem muß für verschiedene Aufgaben einsatzfähig sein, d.h. es sollte mindestens die Forderungen nach automatischer Montage unterschiedlicher Werkstücke erfüllen und dies auch nach 'beliebiger' Reihenfolge. Somit wird deutlich, daß in der Montagetechnik der Begriff Flexibilität zwei Eigenschaften beschreibt:

- Flexibilität nimmt zunächst die Aussagekraft des Begriffs "Vielseitigkeit" an. Je breiter das Werkstückspektrum ist, desto universeller muß das Montagesystem, die Steuerungsauslegung und der verwendete Arbeitsplan sein.
- Die zweite Forderung wertet den Begriff im Sinne der Anpassungsfähigkeit aufgrund eines sich ändernden Werkstückspektrums. Diese Eigenschaft wird um so höher bewertet, je problemloser wechselnde Anforderungen erfüllt werden.

3.1.2 Ausgewählte Aufgabenspektren

Anhand einiger ausgewählter konkreter Flexibilitätsaspekte werden nun Anforderungen und Auswirkungen computerintegrierter Montagesysteme beschrieben (Bild 8).

Automatischer Montageablauf

Zur Verwirklichung ist aus einer Vorgangsbeschreibung Ort, Zeitpunkt und Art der durchzuführenden Fügeoperationen zu entnehmen, d.h. für jedes Werkstück muß zumindest ein ableitbarer Montagevorgang im Arbeitsplan festgelegt sein. Durch den Zellenarbeitsplan sind also die Rahmenbedingungen festzulegen; jedoch darf der Aspekt möglicher Freiheitsgrade (Bearbeitungsstation und zeitliche Reihenfolge) nicht vernachlässigt werden, d.h. der Arbeitsplan darf in erster Linie nur das 'Was ist zu tun' vorschreiben.

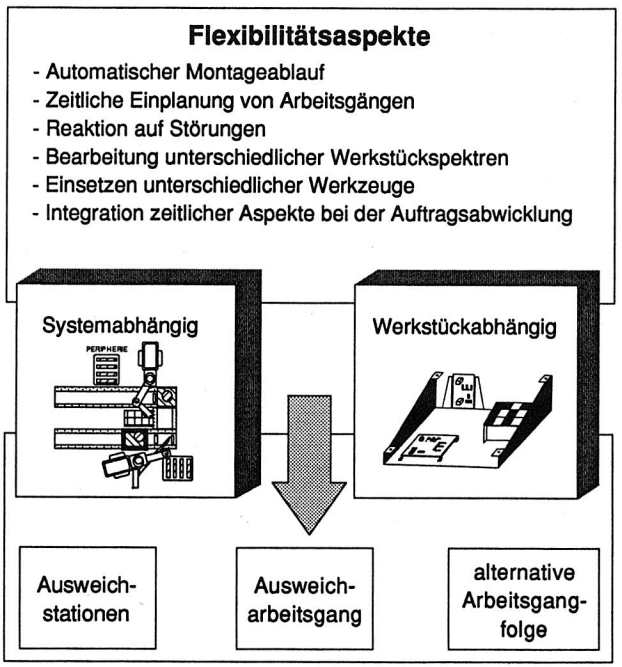


Bild 8: Flexibilitätsaspekte in einer Montagezelle

Zeitliche Einplanung von Arbeitsgängen

Die traditionelle Vorgehensweise plant zuerst die Belegung der Maschinen und richtet anschließend die starre sukzessive Arbeitsgangfolge danach aus (entspricht der Belegungsplanung als vorausschauende Festlegung der in den Stationen abzuarbeitenden Arbeitsgänge mit nachfolgender Reihenfolgeermittlung). Die Reihenfolgeentscheidung geht direkt von der im Verlauf der Belegungsplanung getroffenen Entscheidung über den Fertigungsort aus (frühzeitige Stationsbelegung und Festlegung der Reihenfolge der auszuführenden Arbeitsgänge).

Diese sukzessive Reihenfolgeentscheidung hat also innerhalb einer FMZ keine Gültigkeit, da hierdurch ein gewolltes bzw. notwendiges Ausweichen (Reihenfolgeflexibilität) verhindert wird. Eine Methode der zeitlichen Einplanung der Arbeitsgänge, die die Flexibilität nicht einschränkt und einen 'Automatikbetrieb' ermöglicht, muß deshalb im Planungsvorgang sowohl die Belegungsplanung durchführen als auch die Arbeitsgangreihenfolge festlegen.

Auftretende Störungen

Störungsfälle sind auch bei hoher Zuverlässigkeit aller Systemkomponenten nicht auszuschließen und verringern die aktuelle Verfügbarkeit des Montagesystems. Insbesondere bei hohen Auslastungsgraden läßt sich dann das zeitliche Planziel nur bedingt erfüllen, d.h. bei der im Arbeitsplan vorgegebenen Zeit, die für Disposition, Kalkulation, Kapazitätsberechnung, Durchlaufzeit und für Arbeitsgangterminierung von Bedeutung ist, kann nicht von einer IST-, sondern muß immer von einer SOLL-Zeit ausgegangen werden. Um die Ausfallkosten zu minimieren, genügt es nicht, die Störung schnellstmöglich zu beheben, sondern es muß auch versucht werden, für den Zeitraum der Störungsbehebung einen dem geänderten Systemzustand angepaßten Fertigungsablauf (eingeschränkter Leistungsgrad) zu sichern. Dies erfordert eine schnelle Änderung der ursprünglich festgelegten Arbeitsgangfolge, um den gestörten Bereich zu meiden und eine Produktion im Rahmen des technisch Möglichen aufrecht zu erhalten.

Wechselndes Werkstückspektrum

Die Fähigkeit, verschiedene Montageaufgaben in unterschiedlicher Folge bewältigen zu können, äußert sich auch durch Freiheitsgrade im Ablauf der Arbeitsgänge. Dadurch ist jedoch mit größeren Streuungen der Fertigungszeiten und der Koordinierung von Montagearbeitsgängen zu rechnen, so daß nicht nach langfristig gültigen Fertigungsprogrammen gearbeitet werden kann. Das Ablaufplanungsdilemma tritt besonders bei einer

übergreifenden Produktvariantenfertigung auf (breites Werkstückspektrum). Eine geeignete Arbeitsgangplanungsmethode muß deshalb in der Lage sein, das Reihenfolgeproblem unter der Bedingung wechselnder Auftragszusammensetzungen und sich ändernder Produktspektren automatisch zu lösen. Der Arbeitsplan muß demnach die technische Anpassungsfähigkeit eines Montagesystems durch eine flexible Zuordnung von Stationen und Werkzeugen zu Arbeitsgängen und eine schnelle Reaktionsfähigkeit unterstützen. Zwei Arten von Ausweichmöglichkeiten lassen sich aus der Zuordnung von Arbeitsgängen zu Station und Termin ableiten:

- Ausweicharbeitsgänge und Ausweichstationen
- Alternative Arbeitsgangfolge

Unter Ausweicharbeitsgängen werden Montagealternativen verstanden, die den zunächst eingeplanten Arbeitsgang ersetzen; dies kann dann zur Auswahl einer anderen Station oder auch zu anderen Operationsfolgen führen. Sie lassen sich nur durch Vorabanalyse der werkstückabhängigen Technologie und Geometrie und der im System vorhandenen qualitativen Kapazität ermitteln. Eine Ausweichstation (systemabhängig) ersetzt die ursprünglich vorgesehene Station bezüglich des einzuplanenden Arbeitsgangs. Alternative Arbeitsgangfolgen (werkstückabhängig) lassen sich aus technologisch voneinander unabhängigen Arbeitsgangfolgen ableiten und in der Montageplanung zeitlich entkoppeln, so daß durch Vertauschen solcher Arbeitsgänge aktuell ideale Reihenfolgen zu finden sind. Die werkstückabhängigen Ausweichmöglichkeiten erfordern das Vorhandensein der Fertigungsalternativen im Arbeitsplan. Prinzipiell bestehen hier zwei Realisierungsmöglichkeiten:

- Ein Rahmenarbeitsplan enthält alle Fertigungsalternativen eines Werkstücks.
- Für jede Fertigungsalternative wird ein spezieller Arbeitsplan erstellt.

Die zweite Möglichkeit ist mit einem hohen organisatorischen Aufwand verbunden, da Engpaßsituationen nur durch Ausweichen auf Alternativarbeitspläne beseitigt werden können, wodurch auch durch weitere Engpässe Kettenreaktionen entstehen können (hoher Aufwand zur Generierung). Aus einem Rahmenarbeitsplan, der quasi das Abbild der Arbeitsgangstruktur einer Produktfamilie (mit relevanten Ausweicharbeitsgängen und alternativen Arbeitsgangfolgen) darstellt, kann dagegen in Abhängigkeit des Systemzustandes ein aktueller Arbeitsplan generiert werden (Intelligenz vor Ort notwendig).

Werkzeugfluß

Die vielseitige Einsetzbarkeit unterschiedlicher Werkzeugtypen nimmt vor allem auf Umplanungsmechanismen oder auf Störungsbehebungen Einfluß. Weiterhin bringen

auch alternative Werkzeugbelegungen eine weitere Flexibilitätserhöhung mit sich. Diese Aspekte werden in der Literatur besonders für FFS Systeme in der Vorfertigung /83, 110, 114/ sehr detailliert behandelt. Aufgrund der geringeren Problematik in der Montage können Teilaspekte übernommen werden.

Zeitliche Aspekte

Angaben wie Zeitpunkte und Zeitdauern im Arbeitsplan einer flexiblen Montagezelle haben für unterschiedlichste Funktionskomplexe eine entscheidende Bedeutung, so z.B. für

- Einplanung von Aufträgen (Vorgabe von Sollzeiten)
- Rückmeldung der tatsächlichen Ausführzeiten (BDE)
- Zeitliche Koordinierung von Arbeitsgängen
- Synchronisation von Arbeitsgängen
- Belegungsplanung der einzelnen Stationen
- Wirtschaftliche Gesichtspunkte
  (Kostenberechnung, Investitionsplanung, Instandhaltung)

So bleibt abschließend festzustellen, daß die bessere Ausnutzung der Montagezelle bei häufig wechselnden Kleinserien erst dann wirksam ist, wenn es gelungen ist, die Erstellung und Aufbereitung der Information auf das Leistungsniveau der Montagetechnologie anzuheben.

3.2 Die Arbeitsplanung im CIM Verbund

Die erweiterten Anforderungen an die Steuerungsstruktur von Zellen machen Kenntnisse in organisatorischer, technischer, kaufmännischer und arbeitswissenschaftlicher Hinsicht erforderlich. Den Kernpunkt bildet der Arbeitsplan, der in Flexibilisierungskonzepte integriert werden muß. In der Literatur beschränkt man sich meist auf Fertigungssysteme und läßt die Arbeitsabläufe der komplexeren Montage unberücksichtigt. Um Montagezellen jedoch optimal nutzen zu können, bedarf es einer der Hardware-Flexibilität entsprechenden organisatorischen Arbeitsstruktur. Aus Sicht der Zellensteuerung wurden Untersuchungen über Arbeitsplanstrukturen in konventionellen und zellenspezifischen Arbeitsplänen durchgeführt, um prozeßnahe Anforderungen zu erstellen, die sowohl zum Entwurf einer Arbeitsplanstruktur als auch zur Konzeption logischer Datenstrukturen für Zellenarbeitspläne herangezogen werden.

3.2.1 Arbeitplaninformationen

Der Arbeitsplan schreibt der Montage Art und Ablauf vor. Dafür enthält er alle Angaben, die für eine ordnungsgemäße Abwicklung des Auftrags benötigt werden, d.h. er ist zentraler Informationsträger und soll es auch bleiben, so z.B. für die Kostenrechnung, Materialdisposition und Fertigungssteuerung /28/. Die im Arbeitsplan enthaltenen Informationen lassen sich gliedern in:

- allgemeine Informationen
  (z.B. Identifikationsdaten des APL)
- sachabhängige Angaben
  (z.B. Werkstückklassifizierung)
- vom Arbeitsvorgang abhängige Daten
  (z.B. Arbeitsanweisung, einzusetzende Station/Gerät, Zeitvorgabe)

Die Unterscheidung nach Arbeitsplanarten (Auftragsbezug, Verwendungszweck, Ausführungsform) und daraus resultierende Vorgaben sind in bezug auf die Informationsaufbereitung in der Zelle zu beachten (Integrationseffekt) /40/. Variantenpläne entstehen aus einem Standardarbeitsplan durch Variation der arbeitsgangbezogenen Parameter, wobei die Arbeitsvorgangsfolge beibehalten wird. Aus einem Komplexarbeitsplan kann ein Variantenplan durch zusätzliche Modifikation der Arbeitsablauffolge, z.B. durch Streichen nicht erforderlicher Arbeitsgänge, entwickelt werden.

Die Unterscheidung nach Arbeitsplanungsarten (Arbeitsplanverwaltung, die Variantenplanung, die Anpassungsplanung und Neuplanung) spiegelt sich in unterschiedlichen Planungsprinzipien wider. Wichtig erscheinen in diesem Zusammenhang die Aspekte der Anpassungsplanung und Neuplanung, die grundsätzlich unter Einsatz von werkstückbeschreibenden Systemen und Algorithmen zur Auswahl und Bestimmung der Planungsdaten einen Arbeitsplan erstellen. Hierbei dienen Stücklisten für das jeweilige Erzeugnis als Inputgröße (Wiedergabe des strukturellen Aufbau eines Erzeugnisses oder einer Baugruppe z.B. durch mehrstufige Stückliste).

Die Unterscheidung nach Einflußfaktoren (Einsatzgebiet, Automatisierungsgrad, Fertigungstyp) wirkt sich direkt auf Inhalt und Umfang von Arbeitsplänen aus. So ergeben sich z.B. Unterschiede bei der Arbeitsgangbeschreibung und den Vorgabezeiten durch den Einflußfaktor Automatisierungsgrad. Mit Hilfe einer genau definierten Kinematik und Leistung einer Arbeitsmaschine läßt sich zwar jeder Arbeitsschritt exakt in maschinenverständlicher Form beschreiben, doch zu einer intelligenten Auftragsabwicklung gehören vor allem dispositive Freiheiten.

Beim Fertigungstyp Großserienfertigung findet eine genaue Arbeitsvorgangsermittlung statt und zwar mit dem Ziel, eine kosten- und durchlaufzeitoptimale Lösung zu erhalten. Im Gegensatz dazu können für die Einzel- und Kleinserienfertigung die Fertigungskosten für alternative Verfahren nicht explizit berechnet werden, sondern müssen mit Hilfsmitteln, z.B. Relativkostenkatalogen, abgeschätzt werden. Keine zu starre Zuordnung der Arbeitsgänge (bestimmte vorgeplante Schrittfolge) zu festen Arbeitsplätzen im Arbeitsplan schafft aber die Möglichkeit, auf die momentane Kapazitätssituation aufgrund der Betriebsmittelbelastung und des Auftragspools flexibel zu reagieren und Zeit sowie Kosten zu sparen. Als Beispiel kann die Zuordnung einer Montageaufgabe bzw. deren Arbeitsgänge zu bestimmten Stationen durch die Werkstattsteuerung (Mensch oder / und System) erst bei Auftragsbeginn - auf Grund der aktuellen Situation - herangezogen werden.

3.2.2 Automatisierung der Arbeitsplanung

Die Automatisierung der Arbeitsplanung muß man in zwei Teilgebiete aufgliedern, die Arbeitsplanverwaltung (Bestandteil heutiger PPS Systeme) und die Arbeitsplangenerierung /88/, bei der der Rechner in unterschiedlichen Stufen "schöpferische" Aufgaben des Fertigungsplaners übernimmt. Bei einer FMZ liegt meist Kleinserienfertigung vor, so daß eine Arbeitsplangenerierung - jedoch auf Basisarbeitsplänen beruhend - unter Einbeziehung der jeweiligen Zellendaten sinnvoll erscheint. So kann unter bestimmten Voraussetzungen unmittelbar bei Bedarf mit Hilfe einer gespeicherten Planungslogik auftragsbezogen ein Arbeitsplan erstellt werden.

In diesem Umfeld kann dann die Arbeitsplangenerierung als eine Weiterentwicklung der Speicherung von Variantenplänen bezeichnet werden /40/. Die Auswahlbedingungen können in einer Tabelle zusammengefaßt werden; die Bestimmung der Verzweigungslogik (z.B. Entscheidungstabelle) und der gegenseitigen Abhängigkeiten, die im Rahmen einer Teilefamilie vorkommen, stellt die wesentliche planerische Leistung dar. Die Vorteile einer Generierung von Arbeitsplänen in der Zelle liegen in Anlehnung an /40/ bei:

- der Reduzierung gespeicherten Daten für die Arbeitspläne,
- dem Wegfall des Änderungsdienstes, d.h. es liegen auch im Bereich der Kleinserien stets aktuelle Arbeitspläne vor,
- der Möglichkeit einer Erweiterung und Änderung einer Teilefamilie (Anpassung des Entscheidungstabellensystems),
- der Reduzierung der Planungs- und Erstellzeit und damit der Planungskosten,
- der Erhöhung der Planungsgenauigkeit und der Verbesserung der Dokumentation.

3.2.3 Strukturen für Zellenarbeitspläne

Die Forderung, stets und möglichst schnell auf Basis umfangreicher Planungsdaten und aktueller Zellendaten zu entscheiden und neue Daten erstellen zu können, erzwingt einen strukturierten Aufbau bei Zellenarbeitsplänen. Informationen zu einem Erzeugnis sollen nämlich möglichst auch zellenintern nach klaren Richtlinien gespeichert werden, um nicht nur erhöhte Speicherkapaziät zu vermeiden, sondern auch Datenbasen mit jeweils gleicher Aktualität zu erreichen. Eine Unterteilung nach Arbeitsplankopf (Identifikationsdaten) und -rumpf ist in fast allen eingesetzten Plänen zu erkennen und kann als allgemeingültig angesehen werden. Die wichtigste Anforderung, die den Arbeitsplanrumpf betrifft, ist in Anlehnung an ein Betriebsmittelmodell und Produktmodell nicht nur eine eindeutige Strukturierung der in der Arbeitsvorbereitung einzubeziehenden Daten, sondern auch die Struktur des Arbeitsplanes selbst, um einen schnellen, übersichtlichen Datenaustausch oder -abruf zuzulassen. Arbeitsplanstammsätze sollten in logische Elemente geordnet sein, um für die Aufbereitung neuer Informationen innerhalb der Zelle im wesentlichen Betriebsmitteldaten, Materialdaten, Stations- und Zeitdaten koppeln zu können.

Beim Aufbau eines Arbeitsgangsatzes sind arbeitsfortschritts-, kapazitäts- und belegungsspezifische Probleme zu berücksichtigen (z.B. wichtig bei einer Engpaβbeseitigung), d.h. "alternative Arbeitsgangfolgen" und "Ausweicharbeitsgänge" müssen so erhalten bleiben, daβ diese jederzeit in den Planungsvorgang einbezogen werden können. Auftretende Nachteile einer solchen Strukturierung wie enormer Bedarf an Speicherkapazität, lange Suchdauern und fehlende Transparenz bei komplexen Montagevorgängen können z.B. durch die jeweils aktuelle Betrachtung der alternativen Arbeitsgänge für die nächste Arbeitsfortschrittstufe oder für n folgende (Planungshorizont) eingeschränkt bzw. behoben werden. Ein weiterer Vorteil dieser Vorgehensweise ist, daβ gewählte Alternativen dann im Protokoll abgelegt werden können.

Das systemabhängige Flexibilitätskennzeichen 'Ausweichstation' sollte indirekt über die technologischen Beziehungen des Werkstückspektrums beschrieben werden, z.B. durch die Notation aller möglichen Stationen für einen Arbeitsgang. Änderungen in der Systemkapazität und im Layout können dadurch leichter nachgezogen werden. Die Berücksichtigung dieser Aspekte gewährleistet eine Transparenz des Belegungszustandes der Montagemittel und bietet die Basis, Engpaβsituationen zu erkennen und unter Ausnutzung der Flexibilitätsangaben zu beseitigen. Parametrisierbare Strukturdaten können dann aufgrund von Stammdatensätzen wie z.B. Werkzeugdaten, RC-Programme, Prüfpläne und Belegungs- und Zustandsdaten abgeändert und neu aufbereitet werden.

3.3 Betriebsdatenerfassung

Durch die Verschmelzung der informationstechnischen und der produktionstechnischen Seite entsteht ein Integrationseffekt, durch den der Begriff und die Konzeption CIM mehr ist als nur die Summe der einzelnen Komponenten. Die Erkenntnis, daß neue Produktionstechnologien für sich alleine stehend nur beschränkte Rationalisierungseffekte haben, und der Erfolg wesentlich davon abhängt, wie effektiv sie organisiert und eingesetzt werden, weist den Weg, der von der reinen Automatisierung der Fertigung hin zur Verwirklichung eines CIM-Konzepts einzuschlagen ist: die Integration der informationstechnischen Seite. Aus diesem Blickwinkel heraus ist es nötig, ein verbindendes Instrument zwischen operativen und dispositiven Bereichen zur Verfügung zu haben. So kann und muß die Betriebsdatenerfassung auf Zellenebene zur Realisierung des Integrationsziels wirksam beitragen.

Im Vergleich zum Schwerpunkt der realisierten Betriebsdatenerfassung (Personal-, Auftrags- und Materialdaten) gewinnt die automatische Erfassung von Qualitäts- und Prozeßdaten erst nach und nach an Bedeutung. Für eine Montagezelle besteht aber die Aufgabe, einen Teil zur Realisierung eines Produktionsplans beizutragen, und hierzu gilt es vor allem, die Vorgaben der Produktionsplanung durch eine geeignete Fertigungssteuerung effizient durchzusetzen. Die geforderten Funktionen zur Umsetzung sind aber nur möglich, wenn der Begriff "Betriebsdatenerfassung" unter dem Aspekt 'Rückmeldefunktion' für Planung und Steuerung gesehen wird /45/.

Die BDE wird so als der Teil eines Regelkreissystems angesehen, der die aufgrund von Soll-Vorgaben entstehenden Ist-Daten des Produktionssystems sammelt und der Steuerung ein aktuelles Zustandsbild rückmeldet. Diese kann durch Soll/Ist-Vergleiche die Abweichungen ermitteln und hieraus die Stellgrößen neu justieren /40/. Die BDE schließt somit als Rückkopplungspfad das betriebliche Informationswesen zu einem funktionsfähigen Regelkreis (Bild 9).

Dieser stark abstrahierte Regelkreis setzt sich in der Realität meist aus vielen einzelnen, oft miteinander gekoppelten Sub-Regelkreisen zusammen. Die Schwierigkeit liegt also darin, daß die Betriebsdatenerfassung so konzipiert werden muß, daß die an den verschiedensten "Ecken" auf operativer Ebene anfallenden Ist-Daten kontinuierlich gesammelt und für planende und steuernde Funktionen an anderer Stelle - bei hierarchischen Steuerungssystemen für verschiedene Ebenen - geeignet bereitgestellt werden.

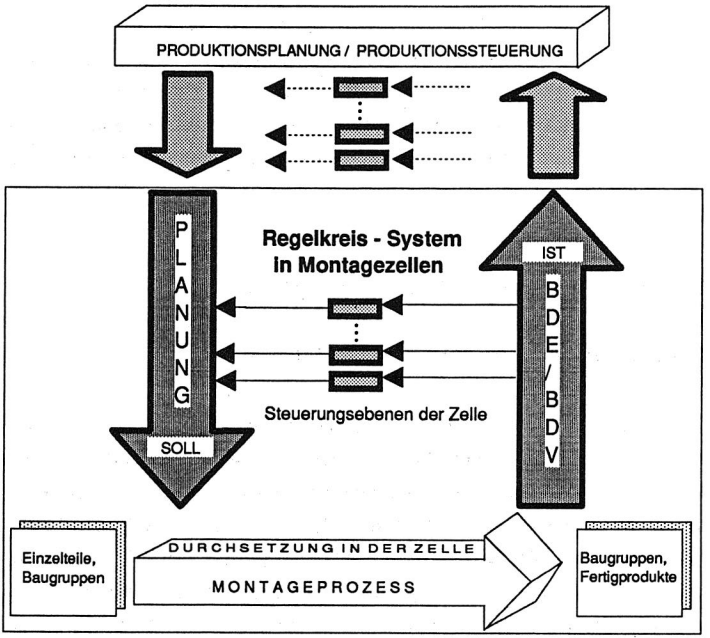


Bild 9: Regelkreissystem in der Zelle durch BDE-Integration

Eine Verbesserung der Steuerungen muß also durch unterschiedliche Technologien bei der Erfassung, Weiterleitung und Verarbeitung der Ist-Daten des Produktionsbereichs unterstützt werden. Anderernfalls geht einerseits die Planung an der Realität vorbei, andererseits kann die kurzfristige Steuerung nicht enger mit den technischen Systemen verknüpft werden. Denn eine Zellensteuerung kann nur dann ihren zeitnahen Aufgaben gerecht werden, wenn sie über eine aktuelle Datenbasis des 'Geschehens' verfügt, d.h. es sollen nur "Ist" Werte und nicht "War" Werte erfaßt und verarbeitet werden. Dies zieht sogleich folgende Forderungen nach sich:

- Die Datenerfassung muß schnell sein, damit aktuelle Daten verarbeitet werden. Denn was nützen schnelle Rechnersysteme, die die Berechnung aufgrund alter und nicht mehr gültiger Daten vornehmen.
- Effektivere Datenerfassung muß durch Datenaufnahme direkt aus dem Fertigungsprozeß oder durch die Datengewinnung aus sofortiger maschineller Verarbeitung erfolgen.
- Eine Basis für Steuerungen müssen genaue Daten sein, d.h. durch automatische Erfassung und Weiterverarbeitung sollen Fehler vermieden und die Zeitdifferenz zwischen Datenentstehung und erster Verarbeitung reduziert werden.

Für die Realisierung eines Zellensteuerungssystems hat dies zur Konsequenz, daß nur eine Gesamtbetrachtung der "Soll" und "Ist" Daten bei den Steuerungskomponenten die Integrationsanforderungen im CAM Bereich (Analyse nach der benötigten Art und nach ihrer zeitlichen Aktualität) erfüllen kann. Hierbei sind die unterschiedlichen Basiskonzepte bezüglich Betriebssystem, Kommunikation und Datenbank zu berücksichtigen, um auf aktuelle Werte zugreifen zu können und eine erfolgreiche Integrationsstrategie zu erreichen (Bild 10).

3.4 Anforderungen an Softwaresysteme

3.4.1 Datenhaltung in Montagesystemen

Ein Gesamtkonzept ist jeweils durch ein leistungsfähiges Datenhaltungssystem zu unterstützen. Indem die Daten der verschiedenen Anwendungsbereiche z.B. in einer Datenbank zusammenlaufen, wird der Datenaustausch vereinheitlicht und redundante Speicherung der Informationen vermieden (vgl. Kap. 2); dies trägt nicht zuletzt zur Integration der verschiedenen technischen Bereiche (CAD, CAP, CAM) bei /75/. Die Vielzahl der Anwendungsprozesse, die gleichzeitig auf die DB zugreifen, bringt aber auch Probleme mit der Datenkontrolle, der Konsistenzerhaltung der Daten und der Zuweisung von Rechten an die verschiedenen Benutzer. Die zwei grundlegenden Ansätze zur Bewältigung der Problematik sind das Ubiquitätsprinzip und das Need-to-know-Prinzip /53/.

Beim Ubiquitätsprinzip in der konventionellen Datenverarbeitung existiert ein global gültiges Datenschema für alle, die auf die DB zugreifen. Grundlage bildet das Transaktionskonzept, das die Operationen auf der DB in atomare und dauerhafte Einheiten einteilt. Das Need-to-know-Prinzip entspricht einer Einschränkung des vorher genannten Konzeptes, für alle Anwendungsprozesse wird der individuelle Datenraum auf den wirklich benötigten Umfang beschränkt. Die Aktualität der Daten und eine Konsistenzsicherung orientieren sich jeweils an den Bedürfnissen der Anwendungsfunktionen.

Neben diesen Datenhaltungsaspekten sind meist in einer Zellenumgebung unterschiedliche prozeßnahe Anforderungen (schnelle Systemreaktionen, Prioritätenvergabe, etc.) zu berücksichtigen. Weiterhin sollten gerade zur Spezifikation einer bestimmten Anwendung bestehende Datenmodellierungsmethoden zur Durchdringung des Problemsbereichs und zur Aufstellung logischer Datenschemata herangezogen werden.

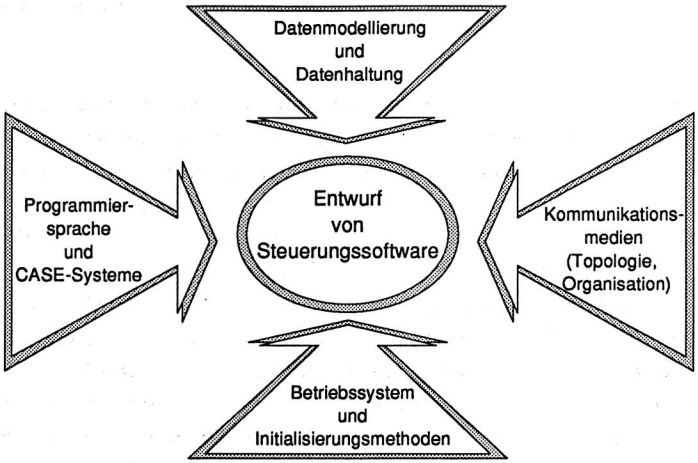


Bild 10: Einfluβ auf die Gestaltung von Softwaresystemen

3.4.2 Betriebssysteme

Als Grundversion für den Einsatz sollte gerade bei komplexen Systemen von Single-User Systemen auf typische Multitasking und Timesharing Betriebssysteme übergegangen werden, um System- und Benutzeranforderungen jeweils optimal zu erfüllen. Innerhalb der Steuerungs-Software unterscheiden sich Prozesse sicherlich hinsichtlich Zeitverhalten und Wichtigkeit. Daher ist es vorteilhaft, wenn Betriebssysteme jedem Prozeβ eine spezifische Priorität mitgeben können, die sich während seiner Lebensdauer nicht verändert. So können Prozesse durch hochpriore Prozesse unterbrochen werden, z.B. zum Zeitpunkt der Zuteilung benötigter Betriebsmittel an den hochprioren Prozeβ. Ein solches Betriebsmittel kann beispielsweise auch eine Nachricht sein, die ein Prozeβ erwartet und schlieβlich von einem Kommunikationspartner erhält.

Unter Konfigurierbarkeit oder dynamischer Erweiterbarkeit des Prozeβ-Systems sollen jene Tätigkeiten zusammengefaβt werden, die mit der Anpassung der zum Betrieb einer flexiblen Montagezelle benötigten Rechner-Hardware und Software an die realen Gegebenheiten vor Ort zu tun haben, so z.B. Verarbeitung unterschiedlicher Planungsmechanismen und Steuerungsstrategien, Anpassung des graphischen Anlagen-Layouts sowie der Bus-Hardware und -Software und der Anpassung des Prozeβ-Systems. Neben der Möglichkeit mittels einer Stapel-Datei als Startmechanismus des Programmsystems sollte

zur Erhöhung der Zuverlässigkeit und Kontrolle auch der Weg über einen zentralen Initiatorprozeβ gewählt werden können.

Aufgrund der rasanten Entwicklung auf dem Hardware-Sektor muβ der Aspekt der Portabilität der Software beachtet werden. Einmal erstellte Software sollte ohne größere Probleme von einer Rechner-Generation auf die nächste übertragen werden; diese Anforderung erfüllt von den Standard-Programmiersprachen vor allem die unter UNIX-Systemen bekannte Sprache C. Unter anderem kann mit der Portierung auf eine leistungsfähigere Hardware eine Erweiterung der Montagezelle realisiert werden, ohne daβ eine Verteilung der Rechnerleistung auf mehrere Rechner mit allen damit verbundenen Problemen (Synchronisation, Schnittstellen, Verwaltungsaufwand) nötig wird. Hiermit eng verbunden ist die Definition von Schnittstellen, insbesondere was die interne und externe Kommunikation angeht. Hierzu sei auf Standardisierungsbemühungen wie das ISO-OSI Basisreferenz - Modell und das auf den Industriebetrieb abgestimmte MAP-Protokoll verwiesen, wodurch in Zukunft eine Veringerung der jeweiligen Kommunikationsproblematik erreicht werden soll /51, 97/.

3.4.3 Kommunikation

Bei einem Steuerungssystem im Zellenrechner müssen grundsätzlich drei Kommunikationspfade unterschieden werden:

- Zellenrechner - übergeordnete Systeme
- Prozeβ - Prozeβ im Zellenrechner
- Zellenrechner - Stationssteuerung

Charakteristische Daten für den Kommunikationsaufwand allgemein in einer FMZ sind einerseits die Anzahl der Stationen (RC-, NC-Geräte) und der durchschnittliche Arbeitsinhalt, andererseits die Funktionalitätsausprägung der Steuerungs-, Datenerfassungs- und Protokollmodule. Zu überlagerten Bereichen (z.B. Leitrechner, NC-Programmiersystem) wird meist eine Bustechnik eingesetzt, d.h. das Steuerungssystem muβ eine Ankopplung an verschiedene Bussysteme (z.B. Token Bus, CSMA) ermöglichen und eine Empfangskomponente bereitstellen, die zunächst unabhängig von der Nachrichtenvielfalt ist.

Aus der Tatsache heraus, daβ Zellen möglichst autonom arbeiten sollen, darf es für einen Leitrechner nicht relevant sein, welches Steuerungsmodul explizit die Daten benötigt (dies hängt von einer konkreten zelleninternen Datenverteilung und Aufbereitung ab). Vielmehr soll er sich jeweils nur mit einem "Ansprechpartner" für bestimmte Datenklas-

sen (NC-Programme, Auftragsdaten) befassen. So wird man dem Informationhiding-Prinzip nach Parnas /74/ gerecht. Legt man das Modulkonzept /112/ zugrunde, so lassen sich drei Stufen unterscheiden, auf denen ein Datenaustausch vorgenommen werden kann: Direktzugriff, Remote Procedure Call (RPC) und die Interprozeβkommunikation (Bild 11).

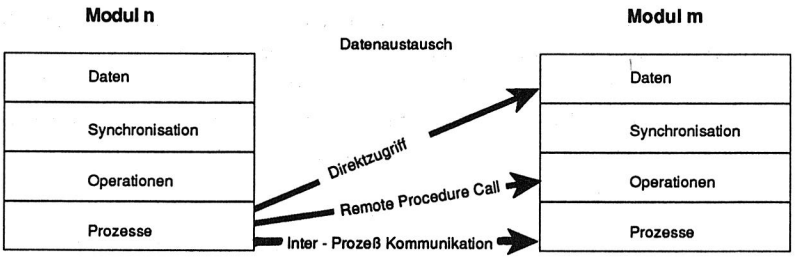


Bild 11: Datenaustausch zwischen Moduln

Der Direktzugriff auf Daten eines anderen Moduls ist einerseits aufgrund schwer lokalisierbarer Fehler und notwendiger Verwaltungsmaβnahmen in komplexen Systemen, andererseits aufgrund der Einsatzmöglichkeiten von Datenbank- bzw. Datenverwaltungssystemen nicht sinnvoll. Trotz Strukturierungsvorteilen mit Hilfe von RPC bleibt hier der groβe Nachteil, daβ der Datenaustausch synchron erfolgt. Die Interprozeβkommunikation kann dagegen synchron und asynchron arbeiten und lehnt sich an die Denkweise der Funktionalitätsaufteilung an, d.h. die Kommunikationspartner sind Prozesse /5/. Damit die Interprozeβkommunikation jedoch effektiv und effizient arbeitet, gilt es, folgendes in die Realität umzusetzen:

- Jeder Prozeβ soll mit jedem anderen Prozeβ kommunizieren können (kein muβ). Beispiel: Zusätzliche Kommunikationspfade zwischen einem Stationsprozeβ und einem BDE - Modul dürfen die Kapazität des Systems nicht sprengen.
- Jeder kommunizierende Prozeβ muβ seine Kommunikationspfade kennen bzw. selbständig herausfinden.
- Es muβ eine Möglichkeit bestehen, einzelne Nachrichten mit Prioritäten zu versehen, um sie vor anderen anstehenden Nachrichten zu selektieren (z.B. Fehlermeldungen).
- Die Länge der zu sendenden Nachrichten soll anpaβbar sein (z.B. Blockgröβen, die von externen Kommunikationsmedien oder der Hardware vorgegeben sind).
- Der Aufbau der Nachrichten soll unabhängig vom aufrufenden Mechanismus sein.

Zusätzlich sollte die Kommunikationsanbindung der operativen Ebene die Möglichkeiten einer Busstruktur oder einer Punkt zu Punkt Kopplung offenlassen.

3.4.4 Initialisierung und Wiederanlauf

Die Produktivität einer FMZ wird wesentlich durch ihr Betriebsverhalten bestimmt. Die Häufigkeit von Störfällen und der Aufwand zur Wiederherstellung der Funktionsfähigkeit sind Optimierungsgrößen, die genauso wie die nominelle Ausbringung zu gewichten sind. So müssen dem Bedienerpersonal Hilfsmittel zur Verfügung stehen, um eine schnelle Fehlerlokalisation zu ermöglichen und aus diesen Angaben Wege zur Fehlerbehebung zu finden. Weiterhin müssen Prozeßinformationen so verarbeitet werden, daß eine vorausschauende Diagnose ermöglicht wird und bei Fehlerfällen eine Recovery durchführbar ist. Unter dem Begriff 'Initialisierung' ist das 'Einfahren' einer Montagezelle ebenso wie die Wiederaufnahme nach einer Unterbrechung zu verstehen. So können durch Betrachtung des Montageablaufs die in Bild 12 aufgezeigten Ursachen der Initialisierung unterschieden werden.

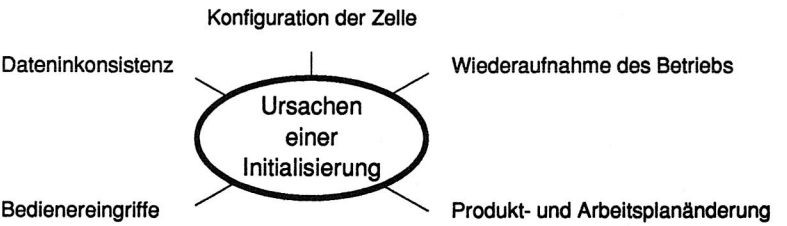


Bild 12: Initialisierungsursachen in einer FMZ

Maßnahmen zur Initialisierung und deren Verlauf sind von den Ursachen abhängig, die zu einer Unterbrechung geführt haben. Aber die Steuerung hat nach Erhalt der Anforderung z.B. Tätigkeiten wie das Beenden der aktuellen Aktionen oder das Sichern von Daten durchzuführen. Ebenso müssen bei Wiederaufnahme bestimmte Maßnahmen durchgeführt werden. Bei einer Unterbrechung der Montage aufgrund einer Blockierung der Steuerung kann je nach Ausmaß der fehlerhaften Daten eine Vielzahl von Änderungen in den Daten oder Aktivitäten in der Zelle notwendig werden, d.h. die Aktualisierung und das Erreichen eines konsistenten Zustands ist zwar schwieriger, sollte aber durch vorhandene Tools unterstützt werden. Mit Hilfe verschiedener Initialisierungsmethoden soll also eine weitgehend vom Bediener unabhängige Wiederaufnahme der Montage ermöglicht werden.

3.4.5 Softwaremodule

Mit zunehmenden Rechnereinsatz und zunehmenden Umfang der zu lösenden Aufgaben wurden auch die Softwaresysteme immer komplexer. Die Entwicklung dieser umfangreichen Systeme mit den auf Intuition und Gewohnheit beruhenden Vorgehensweisen haben SW-Qualitätsziele wie Zuverlässigkeit, Adaption und Portabilität vernachlässigt. Kosten- und Zeitaufwand für die Entwicklung, den Einsatz und die Wartung der Software sind dadurch auch im Fertigungs- und Montagebereich rapide angestiegen /59, 85, 96/.

Diese mit der Erstellung und späteren Anwendung von größerer Software verbundenen Problematik ist Gegenstand der modernen SW-Technologie. Im Mittelpunkt dieser Disziplin stehen Prinzipien, Methoden und Hilfsmittel für die planmäßige, systematische Herstellung komplexer Softwaresysteme. Das Ziel dieser SW-Technologie ist es, durch den Einsatz dieser Verfahren die Softwarekosten zu reduzieren und die Qualität der Softwareprodukte zu erhöhen. Neben den traditionellen softwaretechnologischen Methoden wie strukturierte Analyse, strukturierter Entwurf und strukturierte Programmierung sind hier vor allem neuere Methoden der objektorientierten Architektur mit nichtprozeduralen Spezifikationssprachen -auch für Echtzeitsysteme und parallele Prozesse- richtungsweisend.

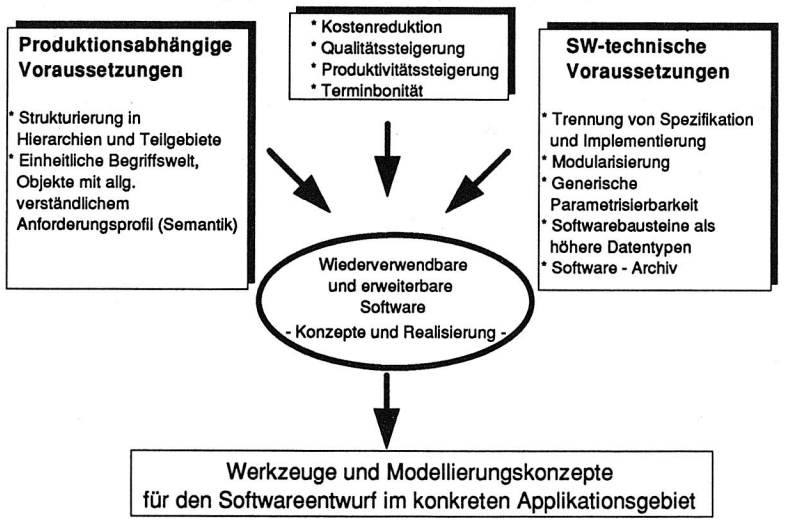


Bild 13: Integrationsaspekte beim Softwareentwurf

Mit Hilfe solcher Sprachen lassen sich Wirkung und Ausführungsbedingungen von Tätigkeiten (Operationen) unabhängig von der Implementierung formulieren. Besonders durch Verbindung mit gewissen Systematiken in bezug auf ein Anwendungsgebiet soll dies eine übersichtliche Aufgabenbeschreibung ermöglichen, so daß sich die Zuverlässigkeit der damit erstellten Software wesentlich erhöht.

Im Anwendungsbereich soll ein Konzept ein adäquates Hilfsmittel zur Modularisierung darstellen und die Realisierung softwaretechnologischer Strukturierungsprinzipien wie Objektorientierung, Abstraktion, Lokalität und Hierarchisierung ermöglichen (Bild 13). Jeweilige Anwender sollten auch vordefinierte und in applikationsbezogenen Bibliotheken gesammelte "Software-Chips" verwenden können und sich so ganz auf die Lösung der eigentlichen Problemstellung konzentrieren /25, 26/.

Um eine gewisse Unabhängigkeit der Module jedoch sicherzustellen, müssen grundlegende Eigenschaften, die sich auf den Konstruktionsaspekt von Softwareprodukten beziehen, berücksichtigt werden. In der Praxis können diese Hilfsmittel aber nur dann verwertet werden, wenn auch die Struktur auf der Anwendungsseite transparent wird und die Zielsetzung in allen Phasen des Entstehungsprozesses mit nachvollziehbaren Kriterien präzisiert wird.

Entwickelte Software sollte so zwar die Anforderungen einer bestimmten Applikation erfüllen, aber als wesentliches Merkmal auch Übertragungsaspekte zulassen. Daher sollte ein Anwender Anforderungen seiner speziellen Applikation mit Hilfe aufbereiteter Softwaremoduln und Generierungsparametern individuell anpassen können, d.h. Softwareelemente sollten (leicht) modifizierbar und vielfach wiederverwendbar sein /53, 76/ und dazu auch eine Möglichkeit bieten, in spezielle Applikationssoftware integriert zu werden.

4 Architekturprinzipien zur Entwicklung der Anwendersoftware

4.1 Grundgedanken

Unabhängig vom konkreten Erscheinungsbild einer Zellensteuerung liegen die Defizite vor allem in der Ausschöpfung der Ablaufmöglichkeiten. Die Probleme beruhen auf unterschiedlichen Informationsständen, mangelnder Transparenz und fehlender Koordinations- und Überwachungsmechanismen. Hierdurch wird nicht nur die Verarbeitung hinsichtlich Fertigungsfortschritt, Qualität und Kosten erschwert, sondern auch auftragsbezogene Korrekturmöglichkeiten werden trotz Planungswissen über Aktivitätsfreiheiten nicht genutzt.

Ein allgemeines offenes Steuerungskonzept für rechnergeführte Montagezellen im Sinne von Integration fehlt, um einerseits beim Aufbau neuer Steuerungssysteme eine Unterstützung zu bieten sowie zugleich schon existierende applikationsverwandte Funktionen zu verwenden und um andererseits Wege zu einer dynamischen internen Auftragsabwicklung aufzuzeigen.

Die Notwendigkeit eines Konzeptes für Steuerungsaufgaben in der Zelle bedeutet, daß ein integriertes Informationsmodell der Auftragslogistik geschaffen werden muß. Ein Ansatz für eine mehrstufige Auftragssteuerung in einer Montagezelle muß demnach mit Hilfe von Modellen Lösungswege und Vorgehensweisen für eine Realisierung aufzeigen. Die Überlegungen müssen sich also von Modellierungsmethoden bis hin zur Schaffung von Softwarefunktionsbausteinen erstrecken (Bild 14).

So sind neben der Übernahme und Bereitstellung auftrags- und betriebsmittelbezogener Daten (von/für Bediener und von/für übergeordnete Systeme) Funktionen für die Verwaltung auftrags- und betriebsmittelrelevanter Daten notwendig. Die Steuerungslogik ist dann so zu gestalten, daß aus Definitionen und Kombinationen neutraler Funktionselemente jeweilige Auftragsdurchlauffolgen entstehen. Die einzusetzenden Überwachungsfunktionen und Statusverwaltungen müssen jeweilige Situationen während der Auftragsdurchlaufzeit erfassen und daraus Zustandsänderungen eines (Teil-)Auftrags, eines Werkstückes oder eines Betriebsmittels ableiten. Innerhalb der Auftragsabwicklungslogistik müssen dann in den einzelnen Detaillierungsebenen weitere - entweder noch anstehende oder aktuell zu generierende - Aktionen automatisch angestoßen werden.

Bild 14: Merkmale eines offenen Systemkonzepts

Für Montagezellen-Steuerungssysteme wird nun aufgezeigt, wie eine Aufteilung bezüglich Aufgaben, Funktionen, Daten und Hardwarekomponenten erfolgen muß, um das Ziel 'Steigerung der Anlagenflexibilität' zu erreichen. Die aufgrund der jeweiligen Anforderungen zu konzipierende Zellensteuerung sollte eine vollständige organisatorische und technische Ausführung eines Auftrages in der Zelle erlauben. Ausgangspunkt bilden hierarchische Steuerungssysteme für FMZ (siehe Kap. 2) unter Verwendung digitaler Rechnersysteme (Komponente eines rechnerunterstützten Gesamtkonzeptes für den CAM-Bereich).

4.1.1 Grundlagen der Modellierung

Das wesentliche Ziel der Informationsstrukturierung ist eine geeignete Modellbildung, um den Grad der Komplexität der Systeme zu reduzieren. Um die gesamte Aufgabenstellung besser erfassen zu können, zerlegt man das komplexe Problem in eine Vielzahl von Einheiten. Der Vorteil ist die leichtere Handhabbarkeit der Teilprobleme, die überschaubarer und damit auch einfacher zu modifizieren sind /1, 29/.

Modelle gleich welcher Art und Ausprägung repräsentieren stets das gleiche System unter verschiedenen Sichtweisen, d.h. Bilder sind jeweils Reduktionen des realen Objektes. Innerhalb dieser Grenzen sind Modelle jedoch unverzichtbarer Bestandteil einer 'Technologie', die Informationen als entscheidenden Produktionsfaktor der nächsten Jahre versteht.

Neben den bekannten mathematischen Modellen bietet die Informatik Prinzipien an, mit deren Hilfe komplexe Systeme modelliert, simuliert und letztendlich optimiert werden können. Diese unter dem Begriff 'Datenmodelle' oder 'semantische Datenmodelle' bekannten Ansätze werden vor allem zum Datenbankdesign - dort Entity Relationship Modelle genannt - herangezogen /72, 95/. Unabhängig von der Art ihrer Implementierung bilden diese Modellprinzipien gerade wegen ihrer graphischen Ausrichtung verstärkt ein Kommunikationsmittel zwischen Systemanalytikern und Anwendern.

Unter einem System wird traditionell ein aus Teilen bestehendes abgeschlossenes Gebilde verstanden, dessen Elemente zueinander in Beziehung stehen, aufeinander einwirken und gewissen Einschränkungen genügen. Die Verknüpfung muß methodisch richtig und aus einem zugrundeliegenden Prinzip vollständig deduzierbar sein (Architektur als Kunst der Systeme /27/).

Als Systemelemente werden bei der Darstellung der Steuerungssystematik nicht nur alle elementaren Teile, sondern auch abgeschlossene Untersysteme (aus Systemelementen zusammengesetzte Teile) betrachtet. Entscheidend ist dabei, daß jeweils mit vorgegebenen Prinzipien Systemstrukturen entwickelt werden können. Hieraus erklärt sich auch, daß Systeme mehr sind als die Ansammlung ihrer Elemente, d.h. sie zeichnen sich dadurch aus, daß sie Eigenschaften besitzen, die aus den Eigenschaften ihrer Elemente nicht erklärbar sind, und daß zu jedem System eine Struktur gehört, die von den Beziehungen zwischen seinen Elementen gebildet wird.

Die Erweiterung liegt hier darin, daß neben der Aufbaustruktur eines Systems, d.h. die Beherrschung einer statischen Struktur, auch für den Zweck der Prozeßführung die Ablaufstruktur, d.h. die Beherrschung der dynamischen Eigenschaften eines Systems, betrachtet und beschrieben werden. Bisherige Modellierungsmethoden, z.B. IDEF/SADT /2/, eignen sich zwar zur Beschreibung mit unterschiedlichem Grad der Vollständigkeitsprüfung und Durchgängigkeit von Daten und Funktionen, aber mit keiner der heute verfügbaren Methoden kann ein Ablauf bis zum flexiblen Einsatz von Ressourcen beschrieben werden /73/. Dies bedeutet, daß keine dieser Beschreibungsmethoden allein als Grundlage zur Darstellung einer Steuerungssystematik verwendet werden kann.

In Anlehnung an /2/ müssen die in diesem Umfeld verwendeten Modellierungsmethoden zusammen den Anspruch erheben,

- die Aufbauorganisations- und Ablaufflexibilität der Zelle aufzuzeigen,
- den flexiblen Einsatz ('online' Zuordnung) von Ressourcen zu ermöglichen,
- eine Vollständigkeit aller beschriebenen Daten und Funktionen sowie eine Durchgängigkeitsprüfung aller beschriebenen Objekte zu erreichen,
- eine Simulation auf allen Detaillierungsebenen zu unterstützen,
- die Anpassung an Methoden und Strukturen zu erleichtern,
- das Modell zur Steuerung und Kontrolle verwendbar zu machen, so daß Änderungen durchgängig beachtet werden,
- durch eine Beschreibungssprache die Abhängigkeit von einzelnen Programmlösungen zu verringern,
- eine Trennung von Datendefinitionen, Ablauforganisationen und Algorithmen vorzusehen, damit jeweilige Anwendungsprogramme transparenter und änderungsfreundlicher werden,
- durch strukturierte Vorgehensweise der Modellierung gleiche Datenbasen zu gewährleisten.

4.1.2 Modell für asynchrone Prozeßsysteme

Je nach Zweck und Ziel läßt sich ein System mehr oder weniger detailliert bzw. konkret modellieren. Eine praktikable Ordnung brachte z.B. Rasmussen /81/ durch Beschreibung über Ebenen funktionaler Abstraktion in die Modellierungswelt ein. Die unteren Ebenen beziehen sich auf die spezifische physikalische Welt, während die oberen die Zielvorstellung, die mit dem System erreicht werden soll, darstellen.

Bezieht man dieses Abstraktionsprinzip und die funktionale Beschreibung von Systemen auf die Steuerungskonzeption, so kann man als Systemmodellierer auf das Modell von Hofmann für asynchrone Prozeßsysteme zurückgreifen /49/. Denn aufgrund einer Aufteilung in eigenständige Prozesse kann eine Steuerungskonzeption als asynchrones Prozeßsystem aufgefaßt werden; gleichzeitig ist aus Sicht der vorhandenen Prozessoren eine Beschreibung der Steuerung als Prozessorsystem möglich. Zusätzlich ergibt sich die Möglichkeit, bei der Beschreibung der Zellensteuerung auf Spezifikationssprachen zurückzugreifen, denen formale Modelle wie z.B. bei /54, 64/ zugrundeliegen. Randbedingungen, die durch die vorhandene Hardware (z.B. Anzahl der Prozessoren) oder durch das Betriebssystem mit möglichen Mechanismen zur Synchronisation oder Prozeßkommunikation gegeben sind, werden erst in der Implementierung berücksichtigt. Im Laufe der Entwicklung vom Elementardatentyp zum abstrakten Prozeßdatentyp mit Synchronisationsanteil finden mehrere Abstraktionsmechanismen Verwendung. Für asynchrone Prozeßsysteme unterscheidet man nach /49/ vier Abstraktionsformen (Bild 15).

Bei der **funktionalen Abstraktion** werden einzelne Algorithmen in separate Einheiten, d.h. in Prozeduren abgetrennt. Bei einer Spezifikation ist es notwendig, die Auswirkung der einzelnen Funktionen auf die Objekte zu beschreiben. Die **Datenabstraktion** wird herangezogen, wenn man dieselben Datenanteile (unabhängig konkreter Datenstrukturen) mit gleichen Operationen bearbeitet . Diese Datenstrukturen mit dazugehörigen Operationen faßt man zu einer Einheit, dem 'Abstrakten Datentyp' (ADT), zusammen.

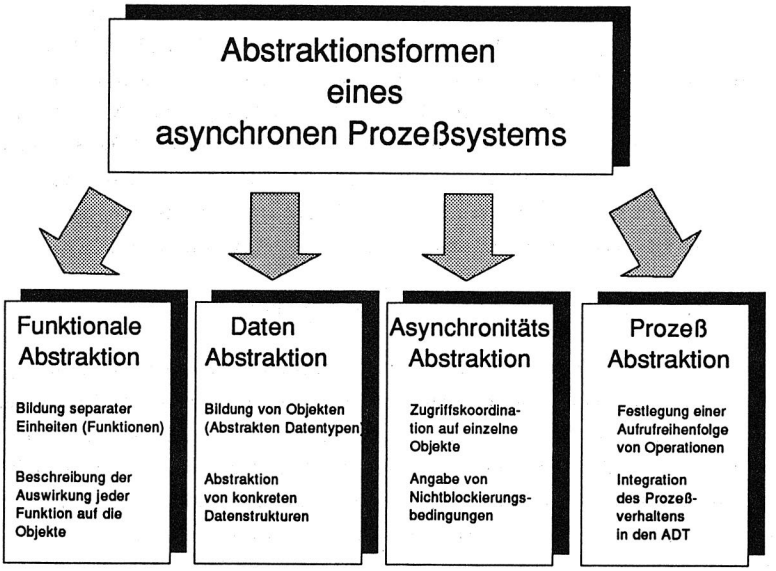


Bild 15: Abstraktionsformen

Ein ADT ist zunächst eine Erweiterung und Modifikation bekannter Datentypkonzepte (z.B. in Pascal); Charakteristika von ADTen werden in Kapitel 5.4 erläutert. Im übertragenen Sinne werden durch dieses fundamentale Konzept der Objektorientierung Objekte (z.B. Puffer, Montageplatz) dem Benutzer zur Verfügung gestellt, wobei jedoch der innere Aufbau (in der Softwareerstellung die Implementierung - in der Fertigungstechnik die Funktionsweise) verborgen wird (Information hiding). Die Daten werden ausschließlich über die zur Verfügung gestellten Operationen manipuliert.

Da bei asynchronen Prozeßsystemen die Abarbeitungsfolge der Operationen (z.B. Überwachungs- und Regelungsaufgaben) im voraus nicht festgelegt werden kann, wurde die **Asynchronitätsabstraktion** eingeführt. Beim Aufruf eines Datenobjekts durch

mehrere Benutzer kann es notwendig sein, die einzelnen Zugriffe zu koordinieren. Da Koordinierungsmaßnahmen an den Schnittstellen zwischen Datentypen auftreten, ist es sinnvoll, diese Synchronisation als Bestandteil in den abstrakten Datentyp aufzunehmen, so daß der Benutzer dieses 'Abstrakten Synchronisierten Datentyps' keine Vorkehrungen für die Koordination der Operationen zu treffen braucht.

Abstrakte synchronisierte Datentypen sind zwar ideale Zerlegungseinheiten (passive Objekte), sie sind aber ohne die Möglichkeit einer zusätzlichen **Prozeßabstraktion** zur Beschreibung von asynchronen Prozeßsystemen nicht ausreichend, da häufig Operationen eines abstrakten Datentyps in einer festgelegten sequentiellen Reihenfolge aufzurufen bzw. passive Operationen in aktive Prozesse (Aktivitätsträger) umzuwandeln sind. Um die übergeordneten Prozesse von diesen Vorgängen, die direkt vom Datentyp und seiner Synchronisation (datentypinterne prozedurale Vorgänge) abhängig sind, zu entlasten, werden dafür notwendige Bestandteile direkt in den Datentyp integriert; durch diese Erweiterung erhält man den 'Abstrakten Synchronisierten Prozeßdatentyp'(PDT).

Formales Modell für einen 'Abstrakten Synchronisierten PDT'

Aufgrund der Zielvorstellungen wird nun eine Technik der Datenmodellierung herangezogen, die die zu modellierenden Objekte und deren Beziehungen in den Mittelpunkt stellt. Ein Objekt kann dabei ein Gegenstand der dinglichen oder der abstrakten Welt sein, so z.B. auch eine Funktion selbst. Die Objekte werden beschrieben durch ihre Eigenschaften (Attribute), die sowohl aus der Welt des Betriebsmittelmodells bzw. des Produktmodells (vgl. Kap.2.3.3) stammen als auch Prozeßeigenschaften darstellen können.

Eine Beziehung stellt dann bekannterweise einen Zusammenhang zwischen zwei oder mehreren Objekten her. Ein Montageauftrag stellt z.B. eine Beziehung zwischen Montagegeräten, Montagevorschriften und Ausführungsdatum her. Durch graphische Darstellungen - Knoten stehen für die Objekte, die Kanten für die Beziehungen - wird bei dieser Art der Modellierung auch die Verständlichkeit für Systemanalytiker, Systemmodellierer und Systemprogrammierer wesentlich erhöht.

Die Begriffe Objekt, abstraktes Objekt, Modul oder abstrakter synchronisierter Prozeßdatentyp werden nun aus Übersichtlichkeitsgründen gleichbedeutend verwendet und entsprechen je nach 'Entwicklungsstufe' einem der obigen Typen.

Dieses formale Modell geht auf die in /54/ entwickelte Betriebssystemmaschine zurück. Jedem Objekt werden die vier Zustandsmengen (laufend, bereit, wartend und blockiert) zugeordnet. In diesen Zustandsmengen befinden sich die globalen Prozesse, die gerade eine Operation auf dem Objekt ausführen oder ausführen wollen. Um für jeden Prozeβ eine systemeindeutige Zuordnung zu einer Zustandsmenge zu erhalten, wird nur "der Teil eines Prozesses betrachtet, für den das abstrakte Objekt verantwortlich ist" /49, 64/, d.h. man betrachtet nur eine Aktivität. Anderenfalls besteht die Möglichkeit, daβ sich ein Prozeβ bei verschiedenen PDTs in unterschiedlichen Zustandsmengen befinden kann.

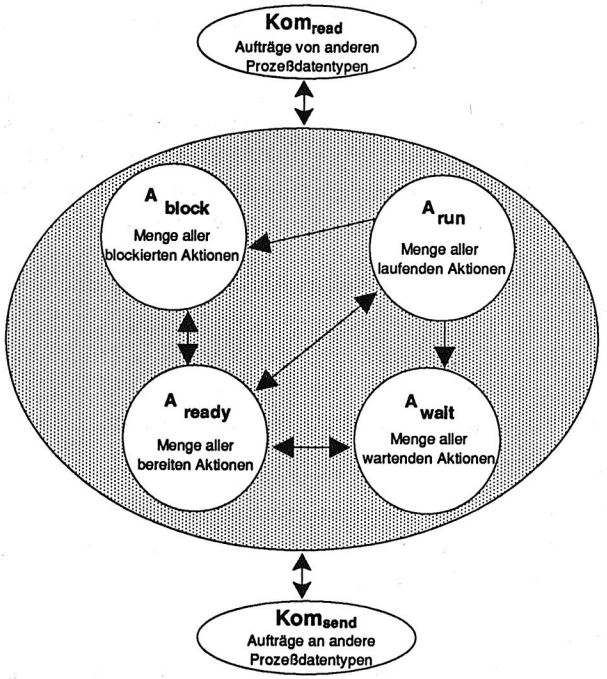


Bild 16: Formales Modell eines abstrakten PDTs

Mit der Wahl einer Aktivität als Einheit wird die Zuordnung in Zustandsmengen eineindeutig. Eine Aktivität wird erzeugt, wenn ein Prozeβ eine Operation des abstrakten Objekts aufruft, sie endet nach Abschluβ der Operation. Durch Einführung des Begriffs der Aktivität ist es möglich, PDTen unabhängig von Eigenschaften globaler Prozesse zu modellieren. Der PDT wird als offenes System modelliert, dessen Operationen zu belie-

bigen Zeitpunkten und beliebig oft aufgerufen werden können. Dies ist ein Grund für die besondere Eignung des Modells zur Darstellung von 'Realzeitsystemen' oder 'der Steuerung einer FMZ'. An die Umwelt wird die Forderung gestellt, daß sie Funktionen zum Transport von Aufträgen und Ergebnissen zwischen den PDTen zur Verfügung stellt.

Die Struktur des Modells für einen abstrakten asynchronen Prozeßdatentyp ist in Bild 16 dargestellt, die exakte formale Definition kann bei /54, 64/ nachgelesen werden. Kom_r bzw. Kom_s sind die Verbindungsstellen des Systems zur Außenwelt. Der Kommunikationsbereich Kom_r (receive) dient zum Ablegen von Anforderungen der Umwelt an das Objekt. Nach Bearbeitung der Aufträge werden die Ergebnisse ebenfalls dort hinterlegt. Im Bereich Kom_s (send) hinterlegt das Modul Aufträge, die von anderen Objekten erledigt werden sollen.

Nach Voraussetzung stellt die Umwelt einen Mechanismus zur Verfügung, der die Aufträge an die ausführenden Module übermittelt und die Ergebnisse zu den anfordernden Modulen zurückbringt. Wichtig ist in diesem Zusammenhang für eine spätere Realisierung auch die Darstellung der möglichen internen Übergänge zwischen den beschriebenen Zustandsmengen und den Kommunikationsbereichen: der Callübergang, der Startübergang, der Requestübergang und der Endübergang.

Der **Callübergang** entspricht dem Aufruf einer Operation und findet jedesmal dann statt, wenn eine neue Anforderung im Bereich Kom_r eintrifft. Eine neue Aktivität wird zur Ausführung des Auftrags innerhalb des Moduls erzeugt und in eine der Mengen A_{block}, A_{wait} oder A_{ready} eingeordnet.

Beim **Startübergang** erfolgt der Wechsel einer Aktivität in die Zustandsmenge A_{run}, vorausgesetzt daß die Anforderungen an bestehende Prioritäten und Verträglichkeiten mit laufenden Aktivitäten erfüllt sind.

Das Modul kann, um Operationen anderer Prozeßdatentypen aufzurufen, Aufträge im eigenen Kommunikationsbereich Kom_s ablegen. Das Ergebnis/Fertigmeldung wird nach Beendigung der Operation dort entnommen. Dieser Übergang wird als **Request übergang** bezeichnet.

Mit dem **Endübergang** findet der Abschluß einer internen Operation bzw. Teiloperation statt, d.h. eine Aktivität wird aus A_{run} entfernt. Nach Abarbeitung wird der zugehörige Auftrag in Kom_r als ausgeführt gekennzeichnet und Ergebnisse dort hinterlegt. Handelt es sich um eine zusammengesetzte oder zyklische Operation, so wird die Aktivität wieder in eine der Mengen A_{wait}, A_{ready} oder A_{block} eingeordnet.

Formales Modell für ein asynchrones Prozeßsystem

Ein asynchrones Prozeßsystem wird i.d.R. aus einer Vielzahl von abstrakten synchronisierten Prozeßdatentypen bestehen, die untereinander beliebig verknüpft sein können, vgl. Struktur in Bild 17.

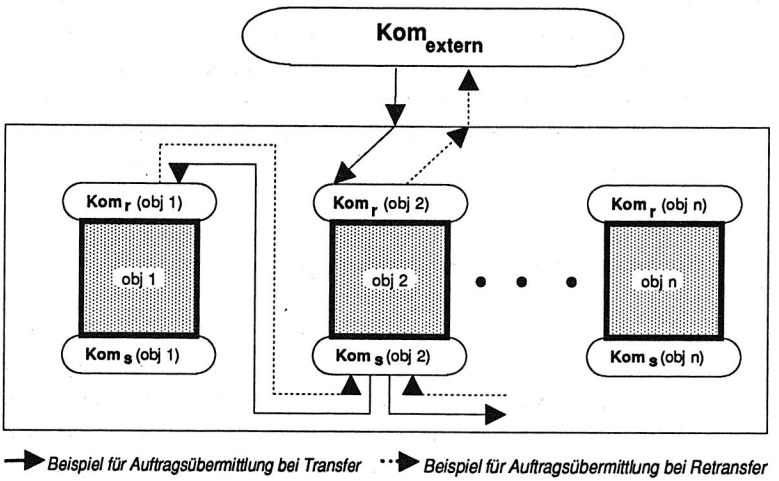


Bild 17: Asynchrones Prozeßsystem

Das Gesamtsystem besitzt einen zusätzlichen Kommunikationsbereich Kom_e (extern), in dem Aufträge hinterlegt werden. Der externe Kommunikationsbereich entspricht in seiner Funktionsweise dem Bereich Kom_r. Er stellt die Verbindung zur Außenwelt dar. Außerhalb des Systems ist nicht sichtbar, welches Modul einen angeforderten Auftrag erledigt, die Zuteilung zu den ausführenden Modulen erfolgt intern. Transfer bzw. Retransfer sind die geforderten Funktionen zum Transport von Anforderungen und Rücktransport der Ergebnisse.
Dieses Modell dient als Grundlage für die Darstellung einer Zellensteuerung. Zur Spezifikation des Prozeßverhaltens der PDTen wird eine an die Syntax und Semantik von Pascal angelehnte Spezifikationssprache von /54, 64/ herangezogen. Die Bestandteile der Sprache bieten den Vorteil, die Spezifikation sowohl auf einem hohen Abstraktionsniveau, ohne Möglichkeiten der Implementierung vorwegzunehmen, als auch auf einer Detaillierungsstufe, die bestimmte Restriktionen der Implementierungsumgebung berücksichtigt, anzugeben. Die formale Beschreibung der Mengen und Relationen des Modells erfolgt bei /64/ auf prädikatenlogischer Basis.

Die Verwendung des Begriffs Prozeß wird innerhalb des Aufbaus eines Steuerungssystems genauso verwendet wie in diesem formalen Modell; d.h wie in /64/ wird auch hier zwischen einem globalen Prozeß und einem Prozeß in bezug auf abstrakte synchronisierte Prozeßdatentypen unterschieden. Ein (system-) globaler Prozeß umfaßt eine Folge von Aktivitäten, d.h. eine Folge von Aufrufen der Operationen, die zu den Prozeßdatentypen eines asynchronen Prozeßsystems gehören. Die Abarbeitungsfolge von Operationen eines asynchronen Prozeßsystems ist häufig vor der Ausführung unbekannt, da das Eintreffen externer Ereignisse, wie z.B. Auftreten von Fehlern oder Eintreffen von Paletten, die Bearbeitung beeinflussen. Ein Prozeß in bezug auf einen PDT wird dagegen verwendet, um aktive Bestandteile eines Objekts zu beschreiben und somit das Prozeßverhalten des PDTs anzugeben.

4.1.3 Kriterien für effiziente Steuerungsstrukturen

Die Konzeption für Montagezellensteuerungen sollte auf einem Schichten-Modell beruhen, das bei der Anwendung Prinzipien wie Abstraktion, Zerlegung und Transformation berücksichtigt und integriert. Erst solche Architekturaspekte bilden eine Basis, daß unterschiedliche Sichtweisen (vertikal und horizontal) in eine Softwarelösung eingearbeitet werden. Bei der Spezifikation und Realisierung komplexer Steuerungssysteme können dann alle Verantwortlichen zur Zielerreichung gleiche Entwicklungswerkzeuge (Beschreibungsmethoden, Softwaretools), fertige Anwendungs- und Kontrollprogramme, aber auch Teilaspekte bestehender Software benutzen. Zur effektiven Erstellung einer individuellen Steuerungssoftware sollte auf bestehende generische Elemente (Basiskonstrukte wie Ringpuffer, Queuemechanismen) und partiale Elemente (Teilkomponenten wie Meldungsverkehr, Stationsmodul, BDE-Modul) zurückgegriffen werden. Erst durch diese Einbeziehung kann in jeder Entwicklungsphase vorhandenes Wissen berücksichtigt und umgesetzt werden. Weiterhin sollte aufgrund der Parallelität zu asynchronen Prozeßsystemen die existierende Entwicklung aus diesem Fachgebiet genutzt werden (vgl. 4.1.2).

Zur Gliederung einer Zellenrechnersoftware (Bild 18) werden neben Dienstleistungsaspekten das Zellenmanagement (Auftragsabwicklung, Verwaltung), die zentrale Zellensteuerung (Ablaufsteuerung), die Montage-/Handhabungs- und Transportsteuerung (Stations-, Materialflußverwaltung) sowie ein Schnittstellensystem zur operativen Ebene (RC, NC, SPS) herangezogen. Ausgehend von Montageaufträgen und dazugehörigen Arbeitsplänen muß durch das Softwaresystem eine Abwicklung gewährleistet werden und zwar bis zum Anstoß entsprechender Handhabungs-/Fügevorgänge und notwendiger

zelleninterner Transporte; die Arbeitsverteilung auf unterlagerte Geräte sollte also dynamisch erfolgen. Neue Teile und Aufträge für Produktfamilien sollten - aus Sicht der Steuerung - jederzeit eingebracht werden können. Der Grundaufbau des Steuerungssystems muß demnach bewußt von einer physikalischen Struktur einer FMZ abstrahieren. Bei Vorlage konkreter Anforderungen und eines Layouts kann dann aufgrund dieser Erkenntnisse und Prinzipien ein Anwendungssystem erstellt werden.

Durch Einführung jeweiliger logischer Elemente in bezug auf Zellenorte, auf Fähigkeiten der Stationen oder Werkstücke wird erreicht, daß höhere Ebenen von technologischen Aspekten quasi unabhängig werden. In 'intelligenten Systemen' wird der gesamte Montagevorgang nicht von vornherein durch feste NC - RC - SPS Programme vorgegeben, sondern bestimmte Grundprogramme werden im online-Betrieb mit Parametern versorgt bzw. gestartet (zustandsabhängige Zellensteuerung).
Die Grundlagen für Steuerungsstrukturen bilden also ein systematischer Aufbau einer FMZ, Aspekte zelleninterner Regelkreise und Arbeitsplanstrukturen sowie Methoden zur Initialisierung, Störungsbehebung und zum Softwareengineering.

Bild 18: Zellenrechnerfunktionalität

4.2 Abstraktion mittels hierarchischem Aufbau

Die sogenannten Schichten - Architekturmodelle bilden die erste Grundlage für eine systematische und einheitliche Konzeption und Realisierung von Steuerungsstrukturen in rechnergeführten Montagezellen sowie für notwendige statische und dynamische Eigenschaften des Steuerungssystems. Durch die Zerlegung des Gesamtsteuerungsablaufs in Schichten wird eine Darstellung der Aufgaben in bestimmten Detaillierungsgraden möglich. Dies wiederum begünstigt die Portierbarkeit von Lösungsansätzen und die Schaffung eines begrifflichen Rahmens für das Applikationsgebiet. Indem der Funktionsumfang der einzelnen Schichten definiert wird, ist es auch möglich, die Abhängigkeiten innerhalb und zwischen den Ebenen (Schnittstellen) zu erfassen. Bezugnehmend auf diese Hierarchisierung der Zellensteuerungsaufgaben können dann auch andere Strukturen entwickelt werden, die eine Basis für die Erfüllung der Flexibilitätsansprüche bilden. Genannt sei zunächst ein entsprechend strukturierter Arbeitsplan, der in jeder Schicht nur bestimmte Elemente für eine Auswertung und zur Generierung von aktuellen Steuerdaten heranzieht. Weiterhin dienen auf diese Schichten abgestimmte abstrakte 'Orts- und Betriebsmittelunterteilungen' der Zelle als Grundlage des Konzeptes.

4.2.1 Grundlagen einer Abstraktion

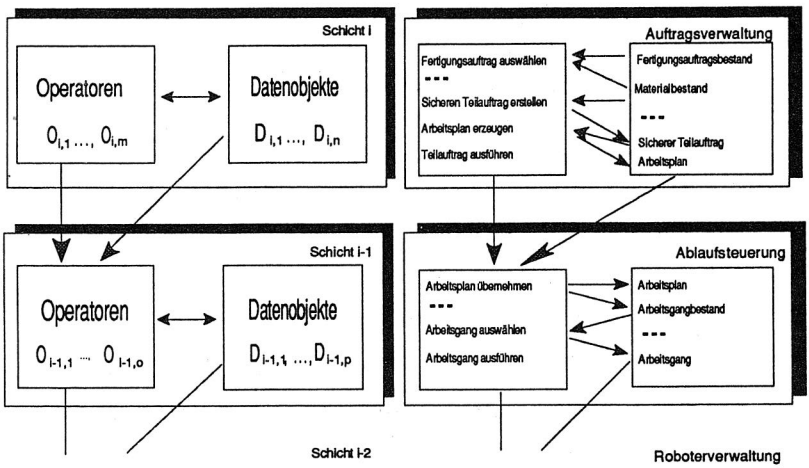


Bild 19: Schicht und Schnittstelle mit Beispiel

Formal besteht eine Abstraktionsschicht i aus einer Menge von Datenobjekten $D_{i,1}$, ..., $D_{i,n}$ und einer Menge von Operatoren $O_{i,1}$, ... , $O_{i,m}$ auf diesen Datenobjekten. Bei strengen Hierarchien darf zur Realisierung der Operatoren einer Schicht i nur auf die Operatoren der darunterliegenden Schicht i-1 zurückgegriffen werden (Bild 19). Wird diese Einschränkung nicht gemacht, so spricht man von einfachen Hierarchien. Die Operatoren einer Schicht i definieren die Schnittstelle i zur Schicht i+1. Die Begriffe Schnittstelle und Ebene werden in diesem Zusammenhang synonym gebraucht.

4.2.2 Hierarchisches Architekturmodell

Im Architekturmodell werden Funktionen jeweils einer Ebene zugeordnet. Die Operatoren der Schichten sind dabei so angelegt, daß sie jeweils ausschließlich mit Hilfe der Fähigkeiten einer bestimmten Betriebsmittelebene realisiert werden. Die Zelle als Ganzes gliedert sich zunächst in 'Stationen' und 'logische Orte', denen in der darunterliegenden Ebene 'interne reale Orte' der Zelle zugeordnet werden. Entsprechend stellt sich der Arbeitsplan als Menge von Arbeitsgängen mit jeweiligen Arbeitsschrittfolgen dar, wobei jeder AS selbst wieder in Einzelaktionen etc. unterteilt werden kann.

Durch diese Gliederung kann jede Funktion einer Schicht als Softwaremodul realisiert und einem Prozeß zugeordnet werden (Ausnutzen möglicher Parallelität der Steuerungsfunktionen). Weiterhin ist es möglich, durch die jeweilige Abstraktion Zustandsinformationen zu den einzelnen Betriebsmitteln erst nach und nach in den Steuerungsprozeß einfließen zu lassen, d.h parametrisierte Steuerfunktionen werden auf der jeweiligen Ebene ergänzt. Dies erleichtert die Reaktion der Steuerung auf Ereignisse und unterstützt auch die Optimierung der Montagevorgänge. Im folgenden werden die aufgestellten sechs Hierarchieebenen zur Steuerung einer FMZ aufgezeigt (Bild 20), und die Konzeptidee der einzelnen Schichten erläutert.

EBENE 6

Die Ebene 6 repräsentiert entsprechend der Strukturierung des Modells die Gesamtmontage in einer Zelle; entscheidend ist also die Aufgabe: Zellenauftrag abwickeln!

Diese Schicht braucht demnach Fähigkeiten zur Übernahme und Verwaltung von Aufträgen, zu Durchführbarkeitsuntersuchungen, zur Stammdaten-, Arbeitsplan- und NC/RC Programmverwaltung, zur Einlastungsplanung und zur Zellenüberwachung. Diese Überlegungen zeigen, daß Funktionen genutzt werden, die den Gesamtzellenauftrag bzw.

das Betriebsmittel Zelle betreffen. Beispiele für Aufgaben der Ebene 6 in konkreten Anlagen sind die belastungsorientierte Auftragsfreigabe in Zellen /113/, Generierung spezieller Arbeitsplan-Ausprägungen, Meldungen an übergeordnete Instanzen (Bestellungen, Störungen, Arbeitsfortschritt, ...).

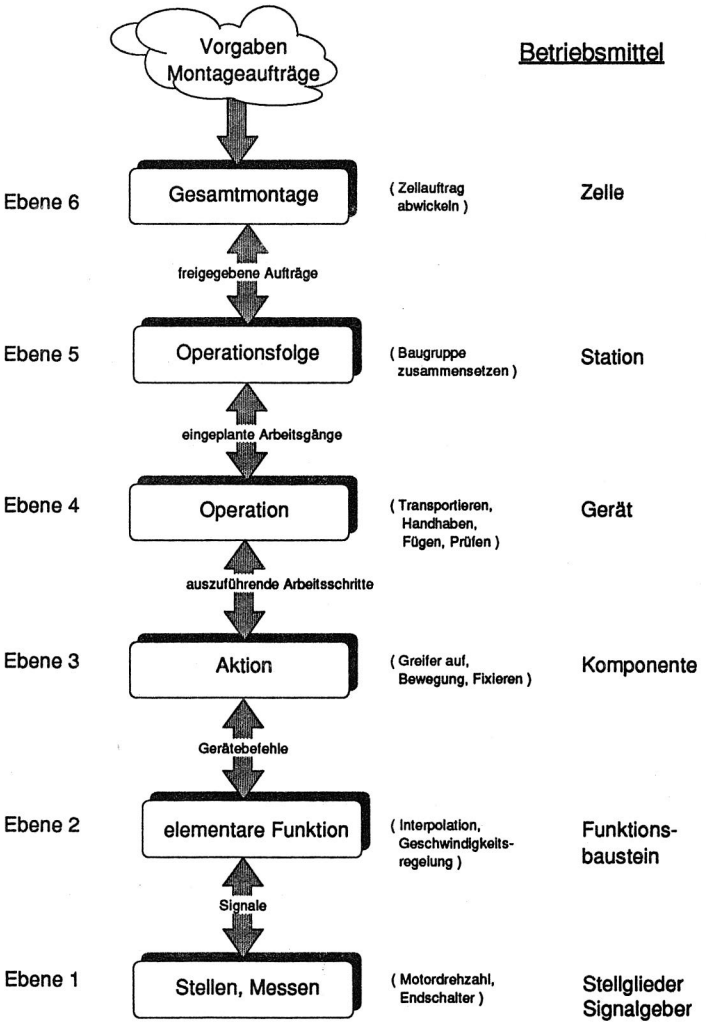


Bild 20: Abstraktes Architekturmodell

EBENE 5

Auf Ebene 5 spielt der Begriff des Arbeitsgangs (zusammenhängende Operationsfolge aus Arbeitsschritten für eine Station) die zentrale Rolle. Ausgehend von Vorgaben des Arbeitsplans erfolgt hier die Auswahl der aktuell zu bearbeitenden AGs und die Zuordnung zu den Stationen, die als Betriebsmittel auf dieser Ebene koordiniert werden müssen. Das System hat zudem die Verfügbarkeit der benötigten Werkstücke an den entsprechenden logischen Bereitstellungsorten sicherzustellen. Die Ebene 5 mit ihren Funktionen bildet die Wirkstelle für die zeitliche und logische Koordinierung der AGs sowie der jeweils daraus abzuleitenden logischen Befehle für das Handhabungs-, Materialfluß- bzw. Werkzeugwechselsystem.

Unter dem Betriebsmittel Station wird maschinenbaulich ein Industrieroboter oder eine NC-Maschine mit Peripherie verstanden. Die Grundlagen für die Synchronisation, Koordinierung und mögliche Parallelität der Arbeitsgänge bilden eine Wissensbasis über Stationsfähigkeiten, die logischen Orte, eine entsprechend leistungsfähige Arbeitsplanstruktur und logische Werkzeug- und Werkstückbezeichner. Von der Zuordnung zwischen logisch ansprechbaren Komponenten wie Bearbeitungsplätze, Übergabeplätze, Puffer, Ein-/Ausgänge und der konkreten physikalischen Realisierung wird abstrahiert.

Diese Ebene -meist Ablaufsteuerung genannt- beinhaltet also in diesem Realisierungskonzept unabhängig von Technologie und Auftragsspezifika Charakteristika und Rahmenbedingungen zur Teilauftragsdurchführung. Hier wird entschieden, daß zur richtigen Zeit die richtigen Teile am richtigen Ort zu einer Baugruppe montiert werden. Besonders die organisatorische Flexibilität wird durch eine freie Arbeitsgangauswahl nach Optimierungskriterien, durch eine aktuelle Zuordnung der Aktivitäten zu den Stationen (online Planung), aber auch durch Erkennen und Vermeiden von Verklemmungen unterstützt.

EBENE 4

Aufgabe der Ebene 4 ist es, die für übergebene Arbeitsgänge benötigten Werte der Arbeitsschrittparameter (entsprechend der 'Freiheitsgrade', z.B. Ortsangaben, Geräte-, Technologiespezifika für das Aufnehmen, Handhaben und Fügen eines Teiles) zu generieren und weitere Schritte (z.B. Teiltransporte aufgrund eines gegebenen Transport-Arbeitsganges, Handhabungsvorgänge bei Austausch bzw. Nachschub von Verbrauchsmaterial) abzuleiten. Die Überlegungen zielen darauf ab, die Generierung von Operationen zu gestatten, deren Parameter erst durch Berücksichtigung der relevanten Teile- und Zustandsdaten erzeugt werden. Voraussetzung für diese quasi 'teilimplizite' RC-Pro-

grammierung ist allerdings, daß die nötigen Geometriedaten der Montageteile und die Peripheriedaten aus der Montageplanung zur Verfügung stehen.

Die Vorgabe der Fügeteile durch logische Bezeichner und der Verzicht z.B. auf die explizite Angabe der Lage eines Teils auf einer Palette und des Fügeortes in Koordinaten ist eine weitere entscheidende Voraussetzung auf dem Weg zu mehr Flexibilität. Es erfolgt dann eine Umsetzung auf die aktuelle Konstellation der Montagestation und auf die darin befindlichen Geräte, d.h. eine Konkretisierung der vorgegebenen Arbeitsschritte. Die eigentliche Fügeoperation (z.B. Fügen durch Einspreizen mit 10 Newton) kann explizit angegeben sein und man erreicht dennoch (für ein bestimmtes Fügeverfahren) einen hohen Grad an Flexibilität und ein "implizites Konzept", das für zukünftige Erweiterungen offen ist.

EBENE 3

In der Ebene 3 gilt es, eine Operation in eine Folge von Aktionen aufzulösen. Diese Trennung und Modularisierung ist in heutigen Roboterprogrammen nicht enthalten. Ziel muß es daher sein, zu generatorischen Verfahren überzugehen /22, 39, 50/. An dieser Stelle im Steuerungsablauf muß die vorgegebene Operation unter Zuhilfenahme einer Datenbasis (z.B. Sensorinformation oder auch nur binäre Zustandsgrößen) in entsprechende Aktionen umgesetzt werden.

Für das Beispiel 'Fügen durch Einlegen' können die Aktionen durch eine Reihe von verfahrensbeschreibenden Parametern bestimmt werden. In der Regel wird in ein feststehendes Basisteil ein Fügeteil eingelegt. Die Art und Weise, wie dies zu geschehen hat, ist durch die Operation gegeben. Die räumliche Beziehung der beiden Komponenten bestimmt im wesentlichen den Gesamt- und Bewegungsablauf. Für das Basisteil können relevante Parameter sein:

- Teilgeometrie
- Teiltoleranzen
- Fügeort auf dem Basisteil
- Fügerichtung
- maximal zulässige Fügekraft

Diese Punkte sind aus der Konstruktion bekannt und können als statische Werte explizit übergeben werden. Zum Teil können diese Angaben aus Materialdaten, Fügeverfahren plus Geometrieinformation sowie Geräte- und Ortsparameter ergänzt und hieraus dann spezifische neue Werte bestimmt werden. Diese Art der Steuerung (Programmierung) läßt sich in verschiedenen Stufen der Auflösung durchführen. Zur Umsetzung der übergebenen Operationen auf konkrete Aktionen der Gerätesteuerungen werden aus den

impliziten RC-Befehlen von Ebene 4 in Abhängigkeit von den Peripheriezustandsdaten die notwendigen Gerätebefehle generiert. Dies ist eine mögliche Schnittstelle zu existierenden Robotersteuerungen. Eine Aktion ist z.B. die Bewegung von einem Start- zu einem Zielpunkt (Befehlsbeispiel: MOVE x, y, z, a, b, c)

Beim Handhaben ist ein Teil in einer bestimmten Lage und Orientierung an den Zielpunkt zu bringen. Im allgemeinen unterliegt der Weg dorthin einer Vielzahl von Randbedingungen und Grenzen, die jedoch für die eigentliche Aktion nicht von Interesse sind. Statt also explizit z.B. alle Hilfs- und Zwischenpunkte oder Beschleunigungen und Geschwindigkeiten festzulegen, sollte auf Funktionen zugegriffen werden, die aus Start- und Zielpunkt und einem entsprechenden Umweltmodell die Bewegung generieren. Ein wesentlicher Bestandteil dieses Modells muß die dynamische geometrische Beschreibung der Umwelt sein, um Kollisionen zu vermeiden und den optimalen Verfahrweg zu bestimmen. Als Output der Ebene 3 entstehen für heutige Steuerungen interpretierbare Befehle. Weiterhin generiert Ebene 4 lediglich Anweisungen, wie Sensoren eingesetzt werden und welche Grenz- und Toleranzwerte einzuhalten sind. Die dynamische Steuerung der damit verbundenen Gerätebewegungen bleibt Aufgabe der Ebene 3. Gerade aufgrund fehlender Standards und unterschiedlicher Implementierungen von Teilaufgaben der Ebene 3 in einer RC-Steuerung, kann in Zukunft eine solche Architektur zur Transparenz dieser Schnittstelle beitragen und die Aspekte der RC Companion Standards bei MMS berücksichtigen.

EBENE 2 und EBENE 1

Diese untersten Hierarchiestufen sind in die Gerätesteuerungen bzw. in die Montagehardware selbst integriert. Hier werden letztendlich die erzeugten Gerätebefehle in Steuersignale umgewandelt. Notwendige Meßgrößen werden umgekehrt von den Hardwareelementen als Signale an die Gerätesteuerung zurückgeliefert, wo sie bei der Befehlsausführung entsprechend berücksichtigt oder zur Ebene 3 weitergegeben werden.
Auf die Realisierung der Funktionsbausteine bzw. der jeweils nötigen Schnittstellen soll an dieser Stelle nicht näher eingegangen werden. Diese befinden sich heute in den abgeschlossenen Robotersteuerungen, werden jedoch im Rahmen von Hierarchiekonzepten bereits neu überdacht /100/.

Im Sinne dieser Hierarchie wird die Richtung der Daten sowohl von oben nach unten als auch umgekehrt betrachtet, d.h. in jeder Ebene existieren nicht nur Vorgaben, sondern es gibt auch Rückmeldungen aus der jeweils unterlagerten Schicht. Die Qualität der zurückgemeldeten Daten weist ein der Ebene entsprechendes Abstraktionsniveau auf (vgl. 4.4).

4.3 Zerlegung der Steuerungssoftware in Komponenten

Der Gedanke der hierarchischen Steuerungsstruktur aus Kap. 4.2 wird im folgenden mit Modellierungsebenen, mit möglichen Ansichten und mit Zerlegungsprinzipien (Trennung der Aufgabenblöcke und Aufteilung auf Steuerungsmodule) verknüpft.

4.3.1 Modellierungsebenen und mögliche Ansichten

Bestehende Modellierungstechniken /2/ arbeiten vorwiegend auf drei Modellierungsebenen. Hierdurch ist die Spezifikationsebene weder eng an die Anforderungsdefinition noch an die Implementierung angelehnt. Durch eine Auftrennung erreicht man vier Modellierungsebenen (Bild 21), von denen jede die Aufgaben einer Zellensteuerung in unterschiedlichen Detaillierungsgraden aufzeigt und die Möglichkeit schafft, einen Bezug zu generischen und partialen Konstrukten herzustellen. Für die Erarbeitung der Funktionen, Daten und Komponenten in den Modellierungsebenen gilt hier der Grundsatz, daß vom Übergeordneten zum Detaillierten (Top-Down-Approach) vorgegangen werden sollte und dies jeweils noch in bezug auf das Architekturmodell.

Modellierungsebene der Anforderungsdefinition

In der ersten Modellierungsebene werden die Anforderungen an das System in Form von Zielen und Einschränkungen definiert. Da auch bei Zellensteuerungen in aller Regel diese Anforderungen von verschiedenen Fachleuten beschrieben werden, ergeben sich ohne Bezug auf ein Rahmenwerk zwangsläufig Redundanzen, suboptimale Beschreibungen und Zielkonflikte. Um die gesetzten Ziele zu erreichen, wird jeweils für einzelne Schichten in der FMZ das 'Was ist zu tun' festgelegt.

Modellierungsebene des Grobentwurfes

In der zweiten Modellierungsebene wird die Anforderungsdefinition umgesetzt in Spezifikationen, d.h. neben Schnittstellenbeschreibungen werden notwendige Datenklassen und Ressourcen festgelegt. Es wird beschrieben, 'wie' und 'mit welchen Mitteln' die Anforderungen der ersten Modellierungsebene erfüllt werden. Die bisher eher globalen, allgemeinverständlich ausgedrückten Festlegungen werden in eine präzisere, mehr technisch orientierte Sprache umgesetzt.

Modellierungsebene des Feinentwurfes

In der dritten Modellierungsebene werden unterschiedliche Aufgaben herauskristallisiert und einzelnen Modulen zugeordnet. Die in der zweiten Modellierungsebene beschriebenen Datenklassen werden also für jede notwendige Funktion in ihre Elemente (Strukturierung der Datenobjekte) zerlegt und mit ihren Beziehungen (z.B. durch Aufstellung von Relationen) aufgelistet. Diese Spezifikation kann sich an Randbedingungen der Implementierung anlehnen (z.B. Layout des Benutzerinterface, Aufgabentrennung Zellenrechner-Stationsgeräte). Nachdem in der ersten Modellierungsebene definiert wurde, 'was' zur Erreichung des Zieles getan werden muß, wird also in der zweiten und dritten spezifiziert, 'wie' die geforderten Ziele erreicht werden. Hierzu wird auf die Methodik der ADTs zurückgegriffen und jeweils die vorgestellten Konstrukte der Spezifikationssprache herangezogen.

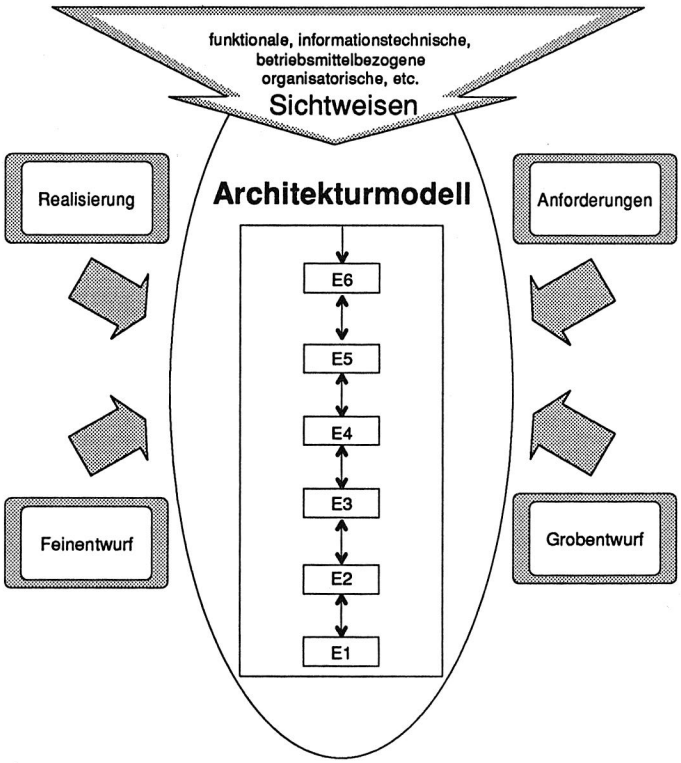


Bild 21: Integration der Spezifikationsebenen

Modellierungsebene der Realisierungsbeschreibung

In der vierten Modellierungsebene werden die getroffenen Festlegungen umgesetzt, d.h. die Aufgaben werden in Modulen und Prozessen realisiert und somit das 'womit' für die Erreichung der geforderten Ziele bestimmt. Als Basis dieser Entscheidung dienen die Software- und Hardwarespezifikationen aus der zweiten und dritten Modellierungsebene einerseits und Beschreibungen von verfügbaren Grundelementen andererseits. Sollten sich die Spezifikationen mit keinen verfügbaren Softwareelementen (generisch und partial) decken, muß entschieden werden, ob neue Module entwickelt oder ob in Iterationen andere Methoden untersucht und spezifiziert werden, die das gewünschte Ziel ohne neu zu schaffende bzw. durch Abänderung bestehender Module erreichen.

Mögliche Ansichten

In einer Architektur sind verschiedene Ansichten zu beachten, so wird in Anlehnung an /2/ unterschieden in Funktions-, Informations-, Ressourcen- und Organisationsansicht.

In der **Funktionsansicht** werden die Aufgaben strukturiert und in bezug auf die Schichten im Architekturmodell beschrieben. Jede der Funktionen, ob global oder detailliert, wird durch einen funktionalen Teil spezifiziert und enthält:

- den Zweck und die Bedingungen, die von der Funktion erfüllt werden müssen,
- die auszuführende Aktivität,
- die Beschreibung der Materialien, Daten, Steuersignale und Ressourcen, die zur Ausführung erforderlich sind,
- die erwarteten Resultate in Form von Materialien, Daten, Steuersignalen und Ressourceninformationen.

Wenn eine Funktion in Unterfunktionen (weitere Detaillierungsebene) aufgelöst wird, erhält die Funktion zusätzlich zum funktionalen Teil noch einen Verhaltensteil mit

- den untergeordneten Funktionen,
- der direkten oder von Konditionen abhängigen Folge der untergeordneten Funktionen und Details der Konditionen,
- dem Zweck und den Bedingungen, die von den untergeordneten Funktionen erfüllt werden müssen,
- den Ereignissen, die die Ausführung der untergeordneten Funktionen veranlassen können.

Im Strukturteil wird das Verhältnis der Vorgänge zueinander beschrieben. Gerade abstrakte Datentypen mit Synchronisationsteil und die Schaffung von Zustandskomponenten können einer Gliederung nach dieser Aufteilung gerecht werden.

In der **Informationsansicht** werden alle Aktivitäten, Daten, Kontrollinformationen, Ressourcen und physischen Objekte der jeweiligen Schicht nach informationstechnischen Konzepten als Objekte beschrieben. Nach Zergliederung in Informationselemente können diese z.B. durch die Entity Relationship - Methode verbunden werden. Externe Darstellungen der Daten (external schemas) werden in möglichst wenige konzeptionelle Schemata (conceptional schemas) für die interne Verarbeitung umgewandelt, und für die jeweiligen Speichermedien in entsprechende Datenformate (internal schemas) umgesetzt.

In der Betriebsmittel- bzw. **Ressourcenansicht** kann die Produktionstechnik, bestehende Rechnerprogramme, Maschinen- und Informationstechnik gesichtet, überprüft, optimiert sowie zugeordnet werden. Die Entscheidung über Anwendungen und Auslastungen der Anlagen kann wiederum durch das Architekturmodell entscheidend unterstützt werden. Bei der Definition der Daten, Funktionen, Kontrollelemente und Ressourcen in den Schichten wird auch festgelegt, wer zu welchen Elementen Zugriffsrecht und Änderungsrecht besitzt. Alle diese Verantwortlichkeiten werden in der **Organisationsansicht** gesichtet und den jeweiligen Funktionen zugeordnet.

Mit diesen Voraussetzungen aus der Modellierungswelt können Komponenten der Steuerung zunächst anlagenunabhängig dargestellt und jeweils einem Baustein einer Schicht zugeordnet werden (Bild 22). Durch Mechanismen des Modul- und Systemdesigns werden diese dann Prozessen zugeordnet und mit Ablaufstrukturen und anlagenabhängigen Datenfiles verknüpft.

4.3.2 Der Modulbegriff als Basis der Zerlegung

Die Komplexität von Zellenrechner-Systemen erfordert, die Modularisierung als Strukturierungskonzept zu verwenden, damit Zusammenhänge noch überblickt und effektive Lösungen gefunden werden können. Beim Zerlegen sollte daher auch ein enger logischer Zusammenhang (z.B. Cohesionsaspekte in der Softwaretechnologie /63/) beibehalten werden. Unter dem Begriff Modulsystem wird eine Menge von Modulen verstanden, die sich innerhalb eines Rechners befindet und ein abgegrenztes Aufgabengebiet bearbeitet. Prozesse bilden die Aktionsträger eines Moduls (vgl . Struktur Kap. 3). Sie bedienen sich der Operationen (Prozeduren), um ihre Aufgabe zu lösen. Die Synchronisation sichert den korrekten Zugang paralleler Prozesse zu den Daten (vergl. Kap. 4.1.2).

Die Strukturierung mit Hilfe von ADTs unterschiedlicher Detaillierungsstufen trägt nicht nur zur Verständlichkeit bei, sondern bietet zusätzlich eine Basis, ein System (zusam-

mengesetzte Komponente) über den Austausch von Komponenten oder durch die Parametrisierung von beteiligten Komponenten zu verändern. Weiterhin wird eine Grundlage geschaffen, einzelne Komponenten durch Anwendung von rechnergeführten Softwareerstellungsmethoden (siehe 5.4) zusammenzufügen (Bild 22).

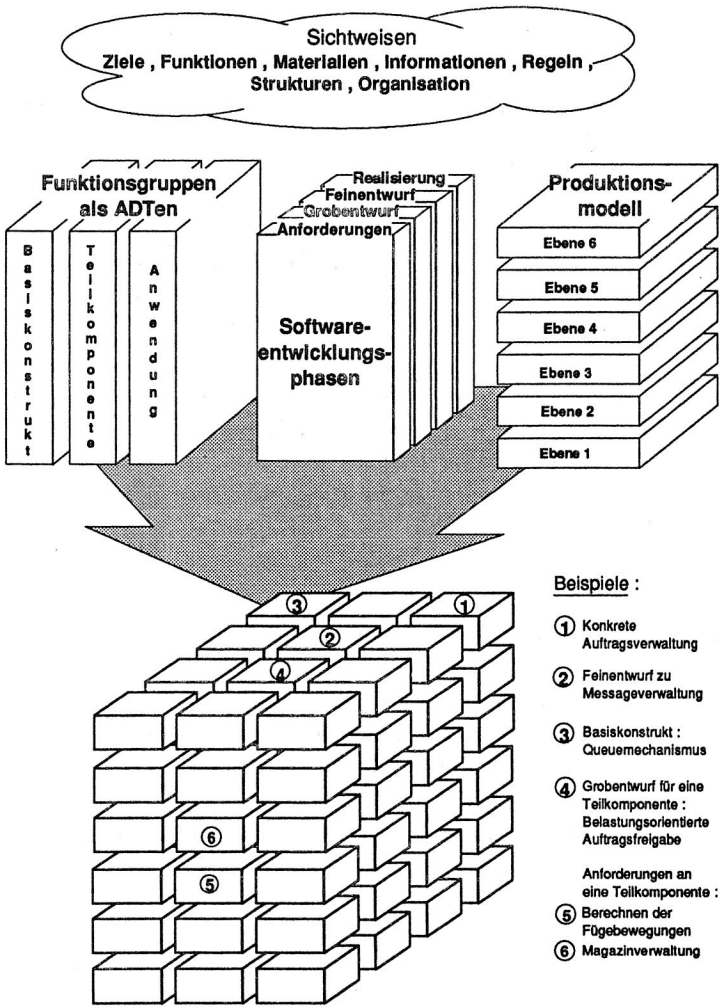


Bild 22: Rahmenwerk zur Gestaltung von Zellensoftware

Die Chance, angebotene Modularisierungskonzepte von Programmiersprachen anzuwenden, wird bisher nur beschränkt genutzt. Gründe sind fehlende Modul-Sammlungen im Anwendungsgebiet, die als Austauscheinheiten verwendet werden können, und der fehlende Einsatz geeigneter Modelle, um Module verständlich zu beschreiben.

Modelle wie die abstrakten Zustandsmaschinen und Prozeßdatentypen sind jedoch insbesondere zur einheitlichen Spezifikation von nebenläufigen Systemen unverzichtbar. Sie liefern Strukturierungshinweise und erlauben erst, Kompositionsoperationen zu identifizieren sowie Relationen zwischen Komponenten verschiedener Abstraktionsniveaus festzulegen. So läßt sich z.B. definieren, wann eine implementierende Komponente eine andere abstrakte Komponente verfeinert.

4.3.3 Ermittlung von Komponenten in einem System

Obwohl für die Ermittlung von Komponenten eines Softwaresystems die Entscheidung meist schwer fällt, etwas in den Mittelpunkt zu stellen, wird - zusätzliche Intuitionen können trotzdem nützlich sein - von folgenden Überlegungen ausgegangen: die zentrale Datenstruktur, die die Dynamik in der Zelle maßgeblich bestimmt, stellt der Auftrag dar. Im VDI Lexikon für Produktionsplanung und Steuerung /108/ kann zum Thema Auftrag folgendes nachgelesen werden:

```
"Schriftliche oder mündliche Aufforderung einer befugten
 Stelle eines Unternehmens zur Ausführung einer Arbeit."
```

Gewiß zwingt die elektronische Datenverarbeitung zu eindeutigeren Definitionen, aber gerade der Aspekt der Anweisung zu einer Ausführung kann übertragen werden: zunächst die Anweisung an eine Zelle, einen Fertigungsauftrag auszuführen, und dann daraus resultierende interne Anweisungen.

Der automatische Auftragsdurchlauf setzt voraus, daß der Integrationsaspekt der Daten vorhanden ist, d.h. es muß die Möglichkeit existieren, Daten wie Arbeitsplan, Geometrie von Teilen, Zustände der Geräte, usw. durch eine Auftragslogistikkette zu verbinden. Ausgehend vom Auftrag und dessen Zuständen werden die Komponenten einer Zellensteuerung dargestellt, d.h. die Aufgaben auf dem Weg vom Zustand bekannt bis zum Zustand fertig werden zugeordnet (fertigungsauftragsorientierte Betrachtung). In der Analyse ist nur von Interesse, 'WAS' für die Ausführung der Anforderungen (Funktion) benötigt wird, das 'WIE', die Ablauffolge und Strategien bleiben im Hintergrund (vgl. Beispiel in 6.4.3).

Es erscheint sinnvoll, die Komponenten in TOP-DOWN Manier darzustellen, um neben der Komponentenzuordnung zu den Ebenen der abstrakten Funktionshierarchie zugleich aufzuzeigen, daß ausgehend vom Fertigungsauftrag Subanforderungen immer durch Erfordernisse der Ausgangsanforderung entstehen. Betrachtet man die Auftragsabwicklung , so erkennt man die Hauptzustände, die ein Fertigungsauftrag durchläuft. Die Aufgaben, die durch den Auftrag "Zellen-FA ausführen" entstehen, kann man in die Blöcke, die in Bild 23 genannt werden, gliedern.

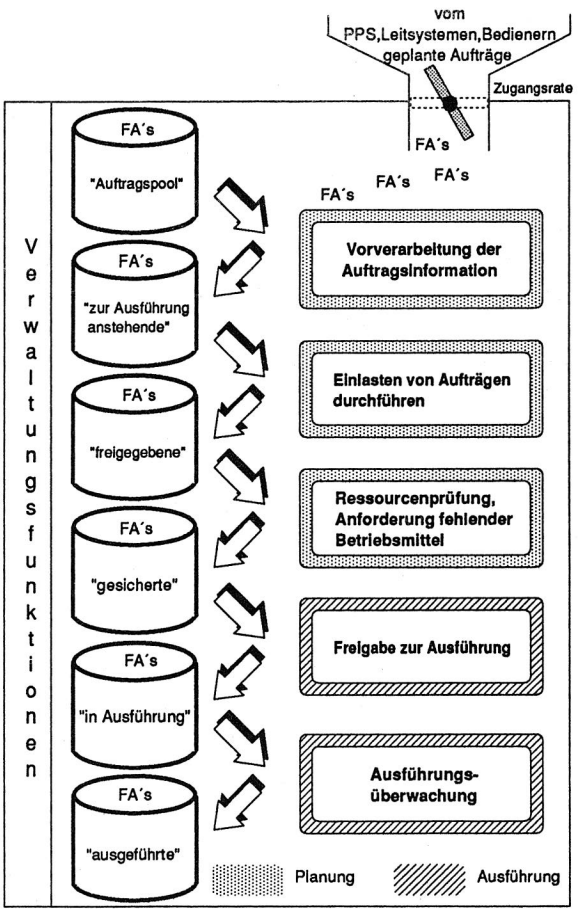


Bild 23: Auftragsabwicklung

In jeder Schicht werden die Aufgaben jeweils in planerische Aspekte und auszuführende Tätigkeiten unterteilt. Die Arbeitsgangabwicklung bildet hinsichtlich der Flexibilisierung des Gesamtsystems den Kern einer Zellensteuerung (vgl. Bild 24). Sie nimmt die Arbeitsgänge für Teilaufträge entgegen, sorgt für deren koordinierte Zuweisung zu Stationen und für die Sicherstellung der Ressourcen-Verfügbarkeit. Als Ergebnis werden parametrisierte Arbeitsschrittlisten für auszuführende Arbeitsgänge weitergegeben.

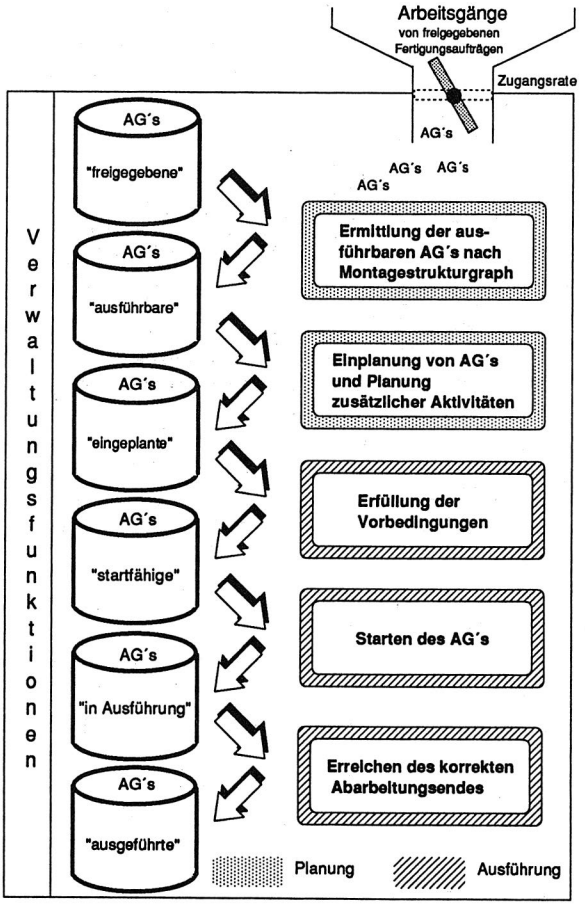


Bild 24: Arbeitsgangabwicklung

Für die Umsetzungstätigkeiten der logischen Arbeitsschritte finden ebenfalls logische und physikalische Aspekte zum jeweiligen Erreichen der Zielzustände Berücksichtigung (Bild 25). Die funktionale Beschreibung der Aufgabe zur Zuordnung von Funktionsträgern ('Fertigungsschritte nach VDI 2860) ist aus dieser Betrachtung (unabhängig von gegebenen Freiheitsgraden) durch Integration von Layoutbeschreibungen und Vorgangsparametern (Geometrie- und Technologiemodell aus der Fertigungsplanung) sowie dem Zustandsmodell erreichbar.

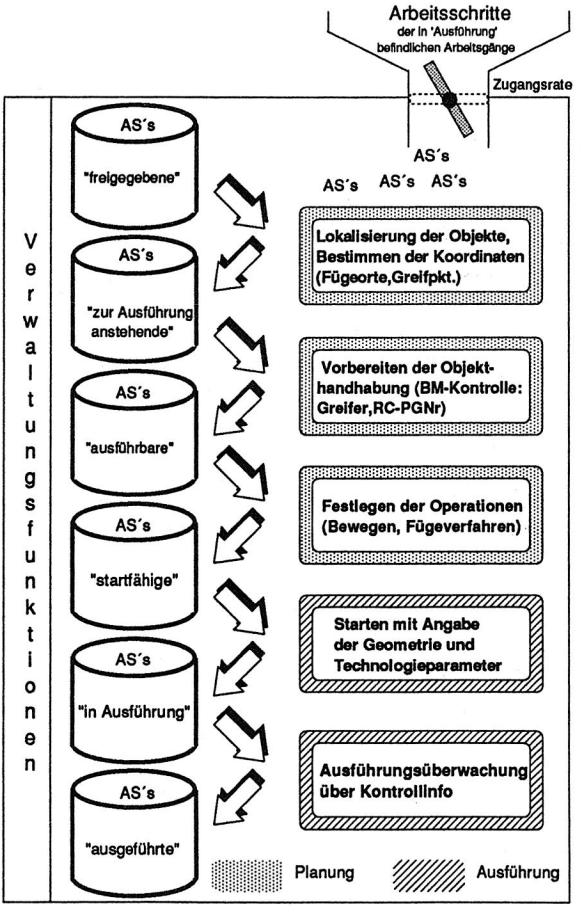


Bild 25: Arbeitsschrittabwicklung

4.4 Prinzip des Regelkreises in der Zelle

Das Abstraktions- und das Zerlegungsprinzip aus den vorhergehenden Abschnitten eignen sich besonders für die Strukturierung statischer Systemeigenschaften. Durch die Tendenz "Flexibilität statt Spezialisierung" sind mehr und mehr dynamische Systemeigenschaften (vgl. die Aspekte der Koordination und Synchronisation in Kap. 6) gefordert, d.h. es ist ein weiteres Prinzip der Modellierung notwendig: das Transformationsprinzip, dargestellt durch eine ereignisorientierte Modellierung z.B als Tripel (Zustand (vorher), Ereignis, Zustand (nachher)).

In bezug auf Darstellungen von Nebenläufigkeit können die zwei Beschreibungskategorien (auch im Sinne der Petri-Netz-Theorie /84/)

Bestehen eines Zustands, *Geschehen eines Ereignisses*

herangezogen werden. Die Nebenläufigkeit, d.h. zwei Ereignisse sind voneinander kausal unabhängig und stehen weder in der Relation "früher als" noch in der Relation "später als" zueinander, bildet somit ein Basiskonzept. Um jedoch auch die Möglichkeiten der Unabhängigkeit entkoppelter Teilsysteme in gewissen Grenzen ausnutzen zu können, (z.B. bei Störungen), muß auf das lokale Erkennen von Zuständen in der Zellensteuerung eingegangen werden, das dann die Grundlage für neue Anweisungen bildet. Dies führt zu einem Zustandskonzept, d.h. jede Koordination und Auslösung eines Ereignisses kann durch eine Menge von Bedingungen und durch eine Zustandsvariable charakterisiert werden /93/.

Bei diesen Zustandsübergängen können weiterhin unterschiedliche Verhaltensmuster (Regeln), Überwachungsmechanismen (z.B. zeitliche Abbruchbedingungen) oder die Behandlung von Ausnahmesituationen integriert werden. Spezielle Ausführungen hierzu sind in /34/ zu finden. Somit ist zunächst zu reflektieren, welche Systematik notwendig wird, um die notwendigen Objektinformationen bereitzustellen.

4.4.1 Rückmeldungen für Planung und Steuerung

Von der Produktionsplanung und Werkstattsteuerung kann mangels genauer Informationen eine Feinsteuerung nicht oder nur unvollkommen vorgenommen werden. Die Zellenebene muß daher aktiv die Steuerungsaufgaben übernehmen, die direkt an der Ausführungsebene liegen. Die bisherige Philosophie, daß die Planungs- und Steuerungsvorhaben übergeordneter Ebenen lediglich "kommentarlos" nach unten weitergegeben werden und Reaktionen wieder ganz nach oben durchgeschaltet werden, wird zugunsten einer

verteilten Intelligenz auf allen Ebenen verändert, die auf ungeplante Ereignisse so früh wie möglich reagiert und Steuerdaten entsprechend abändert. Dies führt zu einem System von hierarchisch geschachtelten Regelkreisen, in denen Steuerdaten und Betriebs- bzw. Maschinendaten in wechselseitiger Abhängigkeit zirkulieren.

Im vorliegenden Umfeld spielen zum Erkennen und Festlegen von Zuständen Aspekte der Betriebsdatenerfassung und der systematischen Datenaufbereitung eine wichtige Rolle. Allgemein wird die "Betriebsdatenerfassung" in definierende und ergänzende Funktionen und Merkmale unterteilt. Erstere sind erforderlich, um Betriebsdaten in maschinell verarbeitungsfähiger Form bereitzustellen.

Mit den ergänzenden Merkmalen können zum Erfassungsvorgang gehörende Verarbeitungsfunktionen verbunden sein. Unter "Betriebsdaten" werden somit die im Laufe eines Produktionsprozesses anfallenden Daten (definierendes Merkmal) und verwendeten Daten (ergänzendes Merkmal) verstanden, wobei es sich um technische und organisatorische Daten handeln kann, vgl. allgemeine BDE - Struktur in einer FMZ (Bild 26).

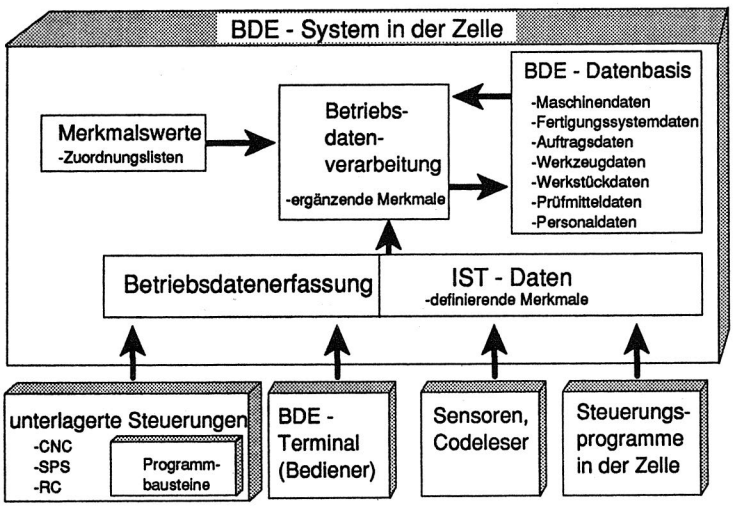


Bild 26: Struktur eines BDE - Systems in der Zelle

Die Bereitstellung von Informationen in maschinell verarbeitbarer Form legt nahe, bei Produktionsprozessen den Einsatz von Sensoren, automatischen Zählern usw. so weit wie möglich zu forcieren bzw. Betriebsdaten direkt aus Ausführungsquittungen von Steueranweisungen abzuleiten. Die übergreifende Aufgabe heißt BDE, BDV und Diagnose unter dem Gesichtspunkt "Einbeziehung dynamischer Aspekte" zu betrachten und zwischen ihnen eine enge Kopplung zu erreichen.

Weiterhin ist anzustreben, daß in der gleichen Weise, in der Soll-Vorgaben (Steuerinformationen) von der Zellensteuerung an die ausführende Geräteebene geschickt werden, über eine gemeinsame Schnittstelle Ist-Daten (Betriebszustandsmeldungen wie Rückmeldungen, Hinweismeldungen, Bedingungsmeldungen und Störungsmeldungen nach DIN 19235) in umgekehrter Richtung bereitgestellt werden.

So muß erreicht werden, daß die im Produktionsprozeß erfaßten Daten unmittelbar zurückfließen, für die jeweiligen Steuerungsebenen geeignet aufbereitet werden und so schnelle Reaktionen auf Änderungen der Peripherie, Maschinenausfälle, Qualitätseinbrüche oder Überlastung von Anlagen ermöglichen.

Daher ist neben den technischen Voraussetzungen eine bestimmte Semantik (Vereinfachung der Inhaltsinterpretation) zur Kanalisierung und Informationsverteilung an verschiedene Interessenten zu schaffen. Die Datensemantik berührt hierbei folgende Bereiche:

- Was: Objektinformation, produktbezogen
- Womit: Objektinformation, betriebsmittelbezogen
- Wie: Vorgangsbeschreibung, verfahrensbezogen
- Wann: Zeitinformation
- Wo(her): Objektinformation, statisch
- Wohin: Objektinformation, dynamisch

Die Integration der informationstechnischen Seite kann weiterhin durch Strukturierungen von Datengruppen (z.B. Aufträge, Betriebsmittel, Material) unterstützt werden. Dies betrifft vorallem Bestrebungen zum standardisierten Informationsaustausch zwischen Automatisierungskomponenten mit MMS (Manufacturing Message Specification) /51/ und mit neueren Entwicklungen wie RC Companion Standards oder PMCS (Production Management Companion Standard) /20, 107/, die die Grundfunktionen (generische Funktionen) von MMS funktional erweitern (Anpassungen an Steuerungstypen). Für die Bearbeitung der Informationsklassen können dann verschiedene MMS Funktionen - unabhängig von der Schichtenabdeckung des Zellenrechners - eingesetzt werden.

Die Betriebs- / Maschinendaten und daraus resultierende Zustandsdaten informieren im Zeitverlauf über Ursache (z.B. ausgeführter Arbeitsschritt, Bedienereingriff) und Art der Veränderungen (z.B. Montagefortschritt, Qualitätsmerkmale) an Objekten. Wichtige Zustandsdaten sollten daher mit Zeitangaben über den Eintritt des Zustands versehen sein, um rückblickend eventuelle Folgewirkungen und Zusammenhänge zu erkennen; diese Logdaten bilden dann eine Basis für Recovery-Aspekte im Steuerungsbereich.

Ein Werkstück kann charakterisiert sein durch eine zeitbezogene Komponente, die den Montagefortschritt erkennen läßt, z.B. eine Liste der ausgeführten Arbeitsgänge und -schritte mit Zeitangaben (Beginn/Ende) und Stationsangaben in tatsächlicher Folge und des aktuellen bzw. des nächsten anstehenden Arbeitsgangs. Eine raumbezogene Komponente läßt erkennen, wo das Werkstück zu finden ist. Je nach Betrachtungsebene kann eine solche Angabe mehr oder weniger detailliert sein. So kann z.B. ein Puffer als 'LO PUFFER_1' angesprochen werden oder direkt ein darin befindlicher realer Ort als 'Platz 14'. Angaben zur Lage können qualitative oder quantitative Angaben sein (z.B. 'Teil liegt richtig/nicht richtig bzw. liegt verschoben zur Sollage um delta(x,y,z)'); ein Bewegungszustand kann z.B. durch 'Teil wird transportiert (von/nach)', 'wird gerade bearbeitet', 'ruht und darf (nicht) bewegt werden' charakterisiert werden.

Eine weitere Zustandskomponente betrifft die Beziehung Werkstück - Montageauftrag ('ungebunden' oder 'reserviert für Auftrag X'). Darüberhinaus kann ein Werkstück aufgrund von Ereignissen (Qualitätsprüfung, Störung bei der Montageoperation etc.) als 'gesperrt' gekennzeichnet werden.

Wichtig auf Ebene 4 sind vorallem die Technologie- und Prozeßdaten (physikalische Größen bei der Durchführung einer Montageoperation). Die Technologiedaten sind hierbei die Soll-Vorgaben für den Montageprozeß, während die Prozeßdaten so zu gestalten und zu verändern sind, daß die vorgegebenen Technologiedaten (z.B. bestimmte Fügerichtung und Fügekraft) erreicht werden.

So sind Prozeßdaten (z.B. Drehmomentüberwachung beim Schrauben und Kraftüberwachung beim Einpressen von Fügeteilen) durch Sensoren oder Meßgeräte zu erfassen, um Abweichungen vom Soll zu ermitteln und sofort korrigieren zu können. Die Überwachung von Prozeßdaten geschieht zwar vorwiegend auf Geräteebene, doch gerade die Sensorik als "Vorstufe der BDE auf der operativen Ebene" und Meldeorgan an die Zellenebene gewinnt zur Vorbereitung neuer SOLL-Vorgaben ständig an Bedeutung /57, 80/.

4.4.2 Der Steuerungs-/BDE-Regelkreis in der Montagezelle

Die Zellensteuerung hat die Aufgabe, die übergebenen Montageaufträge entsprechend der Arbeitspläne und unter Verwendung verfügbarer Resourcen in einer optimalen Weise abzuarbeiten. Jede Steuerungsebene im Schichtenkonzept greift zur Erfüllung der Anforderungen auf die Funktionen der nächsttieferen Schicht zurück und stellt ihrerseits der nächsthöheren Schicht über definierte Schnittstellen ihre Dienste zur Verfügung, wobei der Grad der Abstraktion nach unten hin immer mehr abnimmt /55/. Daneben soll die Möglichkeit direkter Bedienereingriffe gegeben sein, um z.B. Inkonsistenzen (vgl. Kap. 4.6) bereinigen zu können.

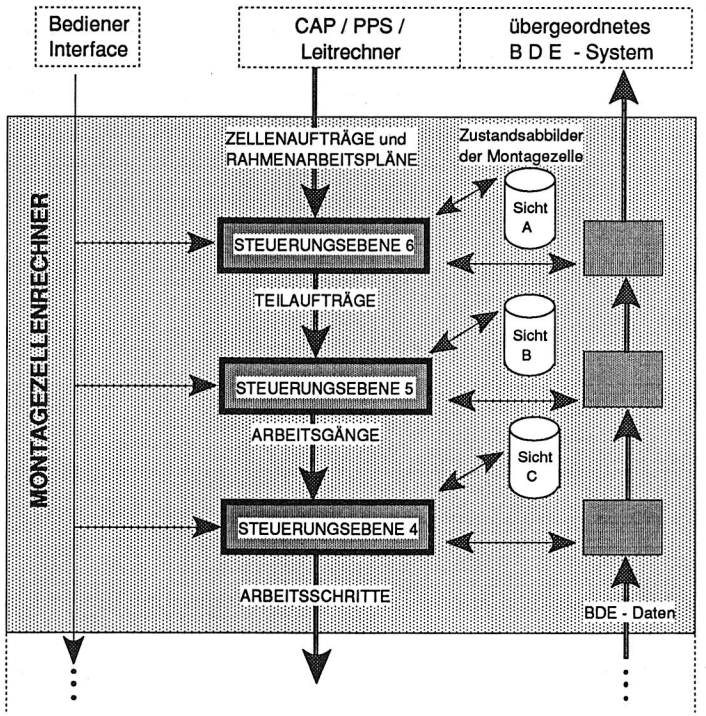


Bild 27: Steuerungs - BDE Regelkreis einer FMZ

Ein Zellenmontageauftrag wird also über die verschiedenen Ebenen der Zellensteuerung in immer detailliertere Steueranweisungen aufgebrochen. In jeder Ebene sind hierbei

Planungs- und Steuerungsfunktionen eingebunden, die auf entsprechende Zustandsinformationen zurückgreifen. Auf diese Weise wird "Intelligenz" auch in tiefere Schichten verlagert und die angestrebte Flexibilität - auch mit einer zeitlichen Reaktionskomponente - wird erreicht (Bild 27).

Grundlegend bei diesem Konzept ist, daß den Montageaufträgen parametrisierte Arbeitspläne zugeordnet werden (vergl. Kap.4.5), die dynamisch auf den einzelnen Ebenen mit Informationen angereichert werden. Die Werte werden aus dem momentanen Zustandsabbild der Zellenelemente abgeleitet, was aber auf den einzelnen Ebenen eine ständige Pflege und Aktualisierung des Datenbestandes bedingt.

Aufgabe eines integrierten BDE-Systems in der Zelle ist es nun, auf die einzelnen Ebenen abgestimmte Daten zur Verfügung zu stellen. Auf jeder Ebene können dann wieder aktuelle Daten in den Steuerdatenstrom einfließen, wodurch ein mehrstufiger Regelkreis geschlossen wird. Beruhend auf unterschiedlicher Verdichtung und Verarbeitung der IST-Daten ergeben sich die im Bild 28 aufgezeigten Datenarten einer Steuerungsebene, die auch unverdichtet oder aufbereitet in höherliegende Ebenen fließen können.

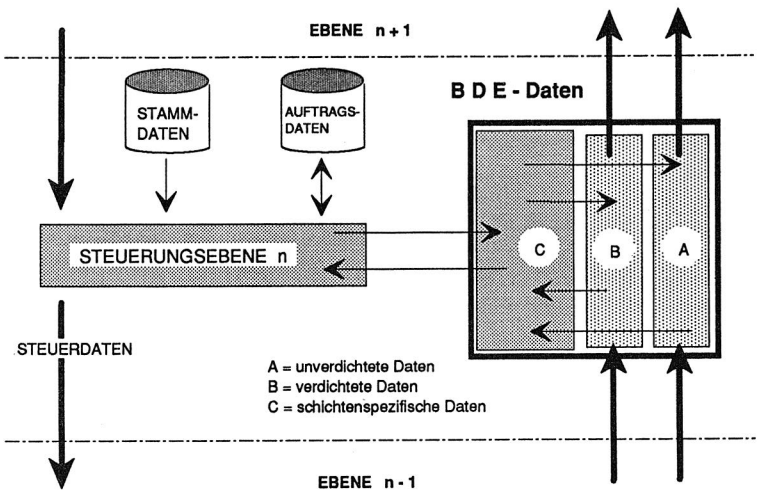


Bild 28: Integration der BDE / BDV in eine Schicht

Zusammenfassend kann festgestellt werden, daß die erfaßten Daten als Zustandsdaten zum größten Teil zellenintern für verschiedene Steuerungsaufgaben, d.h. zum Anstoß von Ereignissen verwendet werden. Darüberhinaus wird jedoch ein Teil der BDE-Daten an übergeordnete Systeme übergeben und einer zellenexternen Verwendung zugeführt. Beim Vergleich der Betriebsdatenerfassung auf Unternehmens- und auf Zellenebene muß schließlich beachtet werden, daß bei zunehmender Annäherung an die operative Ebene vor allem die Automatisierung der Erfassung und die Integration von Verarbeitungsfunktionen in das BDE-System an Bedeutung gewinnen.

4.4.3 Visualisierungsaspekte

Die Prozess-Visualisierung umfaßt das Sichtbarmachen von Sachverhalten, Zuständen und Zustandsänderungen innerhalb technischer Prozesse mit Hilfe eines Rechners. Meßwerte und andere Informationen werden aufgenommen und als Prozeßbilder (z.B. auf einem Monitor) angezeigt. Die Darstellung auf dem Ausgabemedium steht im Vordergrund, da eine schnelle Informationsaufnahme durch den Benutzer erfolgen soll. Statische (z.B. Stationskomponenten) und dynamische Objekte (z.B. Füllstand eines Magazins) zusammen ergeben die Prozeßbilder, die überwiegend aus Hintergrundelementen als Verständnisbasis, Symbolen (konkrete Objekte mit Zustandsinformation), Balkendiagrammen (aktuelle Meßwerte) bzw. Trendkurven (zeitraumbezogenes Abbild) bestehen. Das Einsatzgebiet von Trendkurven liegt in der Montage vorallem bei der Qualitätsprüfung und vorsorgenden Instandhaltung von Anlagen.

Durch Einbeziehung von Grenzwerten und unterschiedlichen Mechanismen (Text, Farbwechsel, Blinken) soll die Aufmerksamkeit auf besondere Zustände bzw. Ereignisse gelenkt werden (z.B. Anweisung "Magazin für Tasten leer - bitte auffüllen!" Die Vorgehensweise bei der Erstellung von Prozeßbildern kann sich an die Zustandsmodellierung anlehnen, d.h. die Aufbereitung geschieht bezüglich der Zustände und möglichen Zustandsänderungen. Die Korrektheit bzw. Aktualität sowie die Schnelligkeit der Arbeitsweise, d.h. das Anzeigen der dynamischen Objekte , ist jeweils vom Prozeßanschluß (Einflußparameter und Zugriff auf diese Daten) abhängig.

Die Gründe für den Einsatz von Prozeßvisualisierung liegen im Bereitstellen von Wissen von Fachleuten für Maschinenbediener und in der begrenzten Informationsmenge, die ein Mensch aufnehmen kann. Deshalb ist es sinnvoll, die angebotene Informationsmenge zu beschränken, gegebenenfalls zu filtern, zu reduzieren und dafür den Informationsgehalt zu steigern. Die optische Darstellung der Vorgänge und Zustände bietet den Vorteil,

einem Menschen pro Einheit die größte Informationsmenge vermitteln zu können und so innerhalb technischer Prozesse einen Eingriff des Bedieners, sei er steuernd oder auch regelnd, zu unterstützen.

Ein weiterer Aspekt ist der Einsatz bei der Entwicklung der Steuerungssoftware, wodurch wiederum durch das Anzeigen von Ergebnissen Zeit und Aufwand gespart werden kann. So wird auch das Testen von Algorithmen bei einer Montageanlage mit Hilfe von PVS erleichtert, da man die Auswirkungen einer Simulation direkt am Monitor verfolgen kann. Die Vorteile eines Einsatzes kann man folgendermaßen zusammenfassen:

- Verkürzung der Reaktionszeit auf gewisse Ereignisse
- Die Gefahr einer Fehlinterpretation wird verringert
- Aussagekraft für verschiedene Betrachter (Sichtweisen!)
- Reduktion der Einfahrphase von Steuerungsprogrammen

4.5 Konzeption einer Arbeitsplanstruktur für Montagezellen

Arbeitsplanstrukturen und dazugehörige Entwicklungen von Datenstrukturen gehören innerhalb der integrierten Datenverarbeitung in der Montage zu den Problemfeldern /78/. Aufgrund von Untersuchungen und Analysen bestehender Arbeitspläne in der Montage (vergl. Kap. 3.2) wird aufgezeigt, wie neben der Erhaltung bewährter Strukturen - zwecks Anwendung bestehender Arbeitsplanerstellungssysteme - neue Elemente zu integrieren sind, um den Anforderungen in einer FMZ gerecht zu werden.
Durch eine Konzeption von Zellenarbeitsplänen soll ein neues Systemdenken erreicht werden, so daß explizit nur das 'WAS ist zu tun' vorgegeben wird. Das 'WANN, WIE und WO ist es zu tun', d.h. von der Planung vorgegebene Freiheiten (Stationsauswahl, Alternativen in der Reihenfolge von AGs, ersetzende AGs), sollen von der Intelligenz der Zelle mitbestimmt werden. Einflußparameter bilden Abhängigkeiten vom Zellenzustand und Optimierungskriterien (Stationsauslastung, Durchlaufzeit, Kostensätze etc.).

4.5.1 Grundgedanken

Das Vorgeben vollständiger Steueranweisungen zur Ausführung eines bestimmten Montageablaufes kann sicherlich keine Basis für die Herstellung vieler Produktvarianten bilden, da weder Zustandsdaten noch Ausprägungen von Baugruppen berücksichtigt werden. Mit Algorithmen und Bedienereingaben sind so unter Einbeziehung eines Auftrages aus 'auftragsneutralen' Montagearbeitsplänen und Erzeugnisstrukturen (Daten aus der CAD/CAP-Welt) jeweils aktuelle Vorgaben zu erstellen (Bild 29).

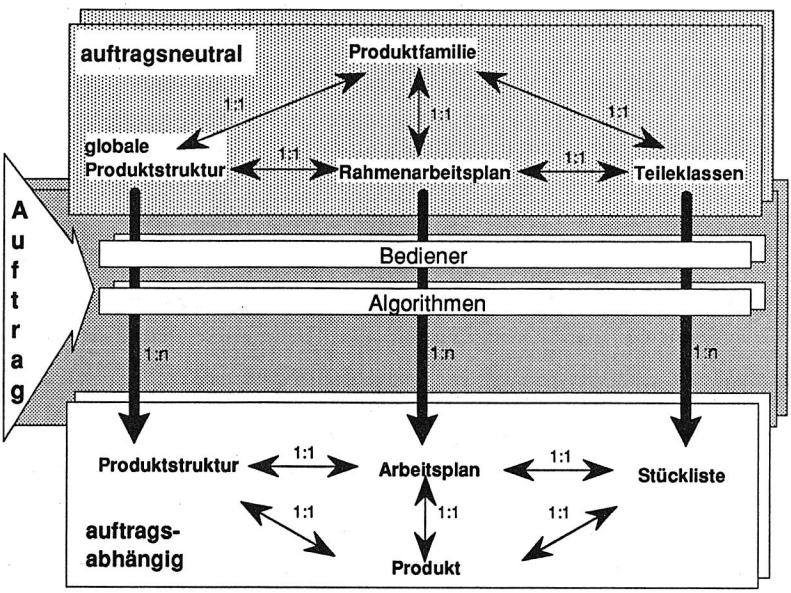


Bild 29: Übergang 'Auftragsneutral - Auftragsabhängig'

4.5.2 Struktur von Arbeitsplänen in einer Montagezelle

Da der Schwerpunkt auf organisatorischen Freiheiten der Zelle und weniger auf Technologiefreiheiten liegt, werden besonders die Auftrags- und Arbeitsgangebene betrachtet. Die Arbeitsschritte werden nur im Hinblick auf die Generierung von Parameterwerten dargestellt, es wird jedoch nicht auf die Aufbereitung von Alternativen und Handhabungsvorgängen eingegangen . So zeichnen eine leistungsfähige Arbeitsplanstruktur zunächst folgende Punkte aus (Bild 30):

- Klare Trennung der Informationen im Arbeitsplan für die einzelnen Schichten des Modells, d.h. jede Ebene sollte nur für sie relevante Informationen aus dem Arbeitsplan ziehen und interpretieren. Die Ebenen 6 und 5 werden so von technologischen Abhängigkeiten befreit.
- Im Arbeitsplan sind keine Transportanweisungen, Be- oder Entladebefehle enthalten. Sie sind von der jeweils verantwortlichen Ebene zu generieren und anzustoßen.
- Der Arbeitsplan ist stationsunabhänig zu halten. Das heißt, für einen Montagevorgang soll im Arbeitsplan bei Alternativen und Ausweichmöglichkeiten noch nicht festgelegt werden, auf welcher Station er ausgeführt wird.
- Ein Arbeitsplan kann je Teilauftrag eine andere aktuelle Ausprägung haben.

- Ein Arbeitsgang ist definiert als eine zusammenhängende Folge von Arbeitsschritten auf einer Station.
- Ein Arbeitsschritt ist definiert als "Pick and Place Operation" für ein Handhabungsgerät mit Operationskennung, Basis- und Fügeteil, Fügeorten und Technologiedaten.
- Jede Ebene besitzt die Fähigkeit ein 'Gesamtpaket' (Arbeitsplan, Arbeitsgang, Arbeitsschritt,...) der übergeordneten Ebene zu übernehmen und zu verarbeiten.

Bezug Auftrag - Rahmenarbeitsplan

Der Begriff Auftrag ist gleichbedeutend mit einer Anweisung an eine Montagezelle ein bestimmtes Erzeugnis x-mal zu montieren. Aufgrund der erzeugnisbeschreibenden Daten ist eine Zuordnung zu genau einer Produktfamilie möglich und die Durchführbarkeit zu testen, d.h. der Auftrag darf keine 'zellenübergreifenden' Informationen vorraussetzen. Erst die Zerlegung eines Auftrags in Teilaufträge für jedes zu montierende Zellenprodukt ermöglicht die flexible Abwicklung, d.h. für jedes einzelne Zellenprodukt kann die Ausführungsreihenfolge der nötigen Arbeitsgänge und auch die Verteilung der Aufgaben in der Zelle nach den aktuellen Zustandsdaten und den Vorgaben durch Rahmenarbeitsplan und Stammdaten frei gewählt werden. Aufbauend auf den Fähigkeiten der einzelnen Montagestationen und der Gesamtzelle liefert der auftragsneutrale Arbeitsplan, genannt Rahmenarbeitsplan (RAPL), die Grundlage für die Beschreibung des Arbeitsablaufs der Montage sämtlicher möglicher Varianten einer Produktfamilie. Jeder Produktfamilie ist genau ein gültiger RAPL zugeordnet.

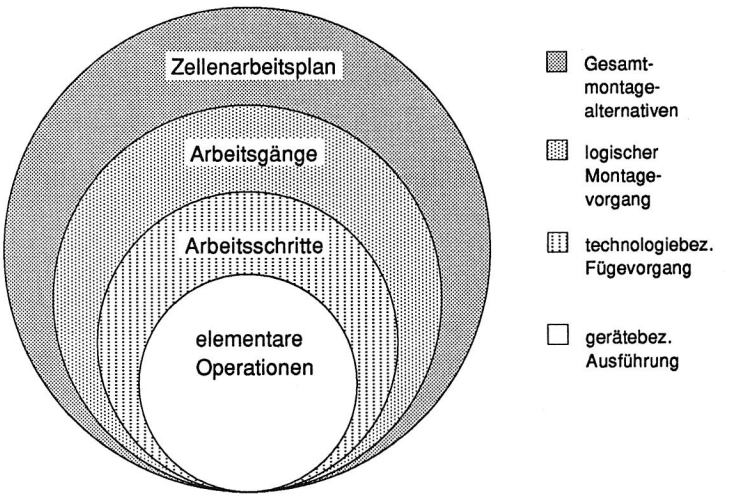


Bild 30: Flexible Zellenarbeitsplanstruktur

Die Daten der RAPL sind unabhängig von der tatsächlichen Ausprägung der Aufträge, so daß der RAPL weder stationsabhängig noch auftragsbezogen ist. Als Darstellungsform für Arbeitsgangfolgen kann eine Netzstruktur herangezogen werden, deren Knoten AGs einer Produktfamilie bzw. einer Produktausprägung sind und deren Kanten die Übergänge von einem AG zu einem anderen darstellen.

Wichtige Informationen des Arbeitsplanes in bezug auf eine AG-Beschreibung betreffen die Stationsunabhängigkeit, Informationen über Teileidentifikation sowie über Ausgangs- und Fertigteile und Angaben über die Koordinierung von AGs. Während der Montage - bedingt durch freie AG-Folgen - muß nicht nur jedes einzelne Werkstück, sondern im Gegensatz zu sequentiellen Abläufen jeder Zustand einer "Baugruppe" nach Ablauf eines AGs individuell identifiziert werden.

Für die Zuordnung von Werkstücken und für den Ablauf des Bereitstellens (Bestellung, Entfernen von Teilen aus der Zelle etc.) müssen Informationen über Teile, die der Zelle zugeführt werden und über Teile, die in der Zelle nicht weiter verarbeitet werden, aus dem Arbeitsplan entnommen werden können. Eine Koordinierung zwischen AGs und eine Koordinierung zwischen einem AG und dem Vorhandensein von Teilen soll nicht allein durch einen Bediener erkannt und bewertet werden, sondern automatisch aufgrund von Arbeitsplaninformationen aufgebaut werden, um jeweils mit maximal möglicher Parallelität zu arbeiten.

Stationsunabhängigkeit des Arbeitsplans

Für eine flexible Abarbeitung muß der Arbeitsplan - soweit möglich - stationsunabhängig aufgebaut werden. Restriktionen sind aber z.B. durch notwendige Angaben über bereitzustellende Teile (und Lagerkapazität) sowie deren Ort für die Ablaufsteuerung gegeben. Die IR-Verwaltung wiederum benötigt stationsspezifische Angaben wie RC-Pgm-Nummern. Somit ergibt sich das Problem, wie die Forderung nach Stationsunabhängigkeit einerseits und die Erfordernis von stationsspezifischen Angaben aus dem Arbeitsplan andererseits zu lösen ist. Zwei unterschiedliche Lösungswege 'Stationsunabhängigkeit durch Fähigkeitsinformationen der Stationen' und 'Stationsunabhängigkeit durch ersetzende Stationen' werden wie die Mechanismen zur Reduktion des Strukturgraphen und zur Abarbeitung eines Graphen auf Arbeitsgangebene in Kap. 6 erläutert.
Insgesamt kann man bei Berücksichtigung dieser Punkte den Arbeitsplan - unabhängig all seiner unterschiedlichen Informationen zur Montageausführung - als Element zur Generierung neuer Steuerinformationen für das Montieren betrachten und als Baustein in eine Zellensteuerung eingliedern.

4.6 Initialisierungsmechanismen, Unterbrechungen, Diagnose

Im Rahmen eines Gesamtprozeßsystems 'Montagezellensteuerung' werden Initialisierungsmethoden erarbeitet und unter Berücksichtigung der vorgestellten Modellierungsmethoden aufbereitet. Unabhängig von einer konkreten Ausprägung eines Gesamtsystems werden die bei der Initialisierung auszuführenden Tätigkeiten ermittelt und insbesondere Recovery- und Resetmaßnahmen analysiert. Für einen abstrakten synchronisierten PDT werden Zustandsdiagramme dargestellt und Zustandsübergänge definiert, die die Grundlage für die Beziehungen zwischen den Zustandswechseln verschiedener PDTen bilden. Der Begriff der Initialisierung umfaßt neben der Erstinitialisierung auch die Wiederaufnahme des Betriebs nach geplanter Unterbrechung und die Behebung inkonsistender Daten des Prozeßsystems durch Recoverymaßnahmen und Resetmechanismen.

4.6.1 Unterbrechungen eines asynchronen Prozeßsystems

Einem Prozeßsystem FMZ und seinen PDTen können eine Reihe von Zuständen zugeordnet werden. Unterbrechungen treten bekannterweise in technischen Systemen zu beliebigen Zeitpunkten ein, das Prozeßsystem und damit jeder seiner PDTen können sich in einem beliebigen Zustand befinden. Da das asynchrone Prozeßsystem FMZ aus den gleichartigen Strukturierungseinheiten 'abstrakter synchronisierter Prozeßdatentyp' aufgebaut ist, ist es ausreichend, alle Betrachtungen auf einen PDT zu beziehen und später auf das Prozeßsystem zu übertragen. Gleichzeitig wird auf Besonderheiten hingewiesen, die sich aus der Tatsache ergeben, daß einzelne PDTen Bestandteile eines Prozeßsystems sind.

Zustandsdiagramm für einen Prozeßdatentyp

Die Steuerungssoftware ist in sich kein abgeschlossenes System, das nur aus abstrakten Aktionen besteht, sondern auch Vorgänge in der Zelle sind von Aktivitäten der PDTen betroffen. Daher können zwei grundsätzlich verschiedene Unterbrechungsursachen ermittelt werden:

- Aufgrund eines Fehlers in der FMZ, d.h im eigentlichen Montageprozeß, kann ein PDT seine Funktionen nicht ausführen und muß bis zum Ende einer Fehlerbehebung (Reparatur) an den Betriebsmitteln der Zelle warten.
- Trotz fehlerfreier Objekte der Zelle kann ein PDT seine Funktionen nicht ausführen, da seine Datenwerte nicht mit dem Abbild der Zelle übereinstimmen, d.h. seine Daten sind inkonsistent. Der PDT ist also blockiert.

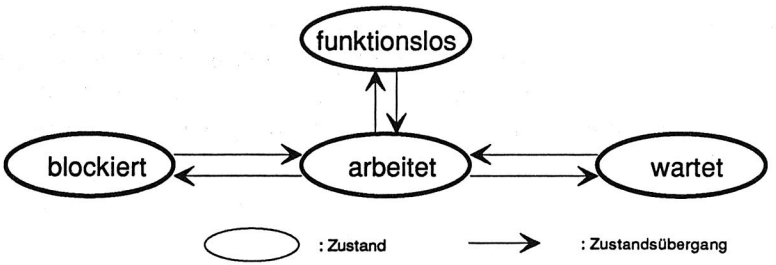


Bild 31: Zustandsdiagramm für einen Prozeβdatentyp (PDT)

Mit Hilfe dieser Trennung kann das Zustandsdiagramm in Bild 31 abgeleitet werden. Ein PDT führt seine anstehenden Operationen aus; er befindet sich im Zustand **'arbeitet'**. Ein PDT kann sich aufgrund verschiedenartiger Ereignisse aktuell auch im Zustand **'wartet'** befinden; Ursachen sind die Durchführung von Reparaturen durch Bedienpersonal oder Unterbrechungen (Pausen) sowie die Beseitigung einer Blockierung eines anderen PDTs. Ein PDT kann keine Funktion ausführen, da er selbst fehlerhaft ist; er befindet sich im Zustand **'blockiert'**. Besitzt ein PDT keine Initialisierungsdaten, so befindet er sich im Zustand **'funktionslos'**.

Zustandsübergänge geschehen durch das Ausführen von Tätigkeiten. Ausgelöst werden Zustandsübergänge durch Ereignisse, die im Ausgangszustand erkannt werden. Nach erfolgreichem Ausführen aller Tätigkeiten eines Übergangs befindet sich der PDT im Zielzustand (vergleiche 4.4).

Da ein Prozeβsystem nicht aus einer Menge von isolierten, sondern von kooperierenden PDTen besteht, genügt es nicht, daβ sich ein PDT nach erfolgreichem Übergang im Zielzustand befindet, sondern bevor er Operationen in diesem Zustand ausführen kann, müssen alle anderen PDTen ebenfalls in einem Zielzustand sein und damit ihren Zustandsübergang abgeschlossen haben. Dies ist vorallem für den Übergang in den Zustand 'arbeitet' von Bedeutung, um die gleiche Sicht auf gemeinsame Variablen und Datenbereiche zu besitzen.

Definition der Zustandsübergänge

Durch Bedingungen, die vor und nach jedem Zustandswechsel gelten, sowie durch die auszuführenden Tätigkeiten wird jeder Zustandsübergang eindeutig definiert. Im folgenden Abschnitt werden hierzu Kriterien, die als Vorbedingung, auszuführende Tätigkeit und Nachbedingung für den jeweiligen Zustandsübergang berücksichtigt werden müssen, aufgelistet.

*Übergang: **funktionslos -> arbeitet***

Vorbedingungen:
- *Ereignis "Systemstart" liegt vor,*
- *Anfangswerte sind in entsprechender Datenbasis vorhanden,*
- *ansonsten gelten die vom System vorgegebenen Anfangswerte (z. B. der Bestand begonnener Arbeitsgänge ist leer),*
- *die Sollwerte (Stamm- oder Technologiedaten) liegen vor,*
- *es existieren keine Quittungen oder Meldungen,*

auszuführende Tätigkeiten:
- *"Systemstart" an andere PDTen weiterleiten,*
- *alle Initialisierungsdaten übernehmen,*
- *Daten für andere PDTen bereitstellen,*
- *benötigte Daten von anderen PDTen entgegennehmen, verarbeiten und intern zuweisen,*
- *Daten auf Konsistenz überprüfen,*
- *in den Zustand 'arbeitet' übergehen und dies weitermelden,*

Nachbedingungen:
- *alle Variablen besitzen ihre aktuellen Anfangswerte,*
- *konsistenter Zustand erreicht, PDT kann Normalbetrieb aufnehmen, sobald alle PDTen den Übergang nach 'arbeitet' abgeschlossen haben.*

*Übergang: **arbeitet -> wartet***

Vorbedingungen:
- *Ereignis "System halt! " (z.B. Eingriff oder Pause),*
- *PDT ist in einem konsistenten Zustand,*

auszuführende Tätigkeiten:
- *Ereignis an andere PDTen weitermelden,*
- *laufende Aktion zu Ende bearbeiten,*
- *aktuelle Zustandsdaten sichern (z.B. in einer Datenbank),*
- *in den Zustand 'wartet' übergehen und dies weitermelden,*

Nachbedingungen:
- *letzter aktueller Datenbestand gesichert,*
- *PDT ist im aktuellen und konsistenten Zustand.*

*Übergang: **wartet -> arbeitet***

Vorbedingungen:
- *Ereignis "Wiederanlauf" liegt vor,*
- *PDT ist in einem konsistenten Zustand,*
- *gesicherte Daten unverändert in der Datenbank vorhanden,*
- *zwischenzeitlich durchgeführte Änderungen sind bekannt und abgelegt, ebenso auszuführende Aktionen vor Aufnahme des Normalbetriebs,*

auszuführende Tätigkeiten:
- *Ereignis "Wiederanlauf" weitergeben,*
- *Aktualisierung der gesicherten Daten (Änderungsdienst)*
- *aktuelle Daten an lokale Datenbereiche übergeben,*
- *geforderte Aktionen ausführen,*
- *Tests auf Konsistenz durchführen,*
- *in den Zustand 'arbeitet' übergehen,*
- *Übergang an anderen PDTen weitergeben,*

Nachbedingungen:
- *aktuelle und konsistente Daten sind vorhanden,*
- *alle gewünschten Aktionen ordnungsgemäß ausgeführt,*
- *PDT ist in einem konsistenten Zustand, bereit zur Aufnahme des Normalbetriebes.*

*Übergang: **arbeitet -> blockiert***

Vorbedingungen:
- *Ereignis "Blockierung" ist eingetreten,*
- *Ausführung des PDTs ist unterbrochen, d. h. die aktuell ausgeführte Aktion ist abgebrochen,*
- *PDT im inkonsistenten Zustand,*

auszuführende Tätigkeiten:
- *Ereignis weitergeben, andere PDTs müssen nach 'wartet' wechseln,*
- *aktuelle Daten - soweit möglich - sichern,*
- *Fehlerursache und betroffene PDTen ermitteln, dazu gehören z.B. auch Zeitpunkt und Ort der Fehlerursache,*

Nachbedingungen:
- *PDT in inkonsistenten Zustand (fehlerhafte Daten),*
- *die 'fehlerfreien' Daten sind gesichert,*
- *Fehlerursache, -zeitpunkt und betroffene PDTen ermittelt.*

*Übergang: **blockiert -> arbeitet***

Vorbedingungen:
- *PDT ist in einem inkonsistenten Zustand,*
- *Ereignis "Recovery- bzw. Resetbeginn" liegt vor,*
- *Beginn der Behebung, sobald alle anderen PDTen selbst im Zustand 'wartet' sind,*

auszuführende Tätigkeiten:
- *Ereignis "Recovery- bzw. Resetbeginn" weitergeben,*
- *aus den aktuellen Daten anderer PDTen und eigenen gespeicherten Daten einen konsistenten Zustand zur Basis des Weiterarbeitens ermitteln,*
- *Datenbasis aktualisieren,*
- *konsistente Daten an lokale Datenbereiche weitergeben,*
- *physische Aktionen initiieren,*
- *Tests ausführen,*
- *in den Zustand 'arbeitet' übergehen und weitermelden,*

Nachbedingungen:
- *PDT in konsistenten Zustand, der nicht mit dem letzten konsistenten Zustand vorher identisch sein muß,*
- *alle Daten wieder konsistent,*
- *Daten an lokale Datenbereiche übergeben.*

Übergang: ***arbeitet -> funktionslos***

Vorbedingungen:
- *PDT in einem konsistenten Zustand,*
- *Ereignis "System aus" liegt vor,*

auszuführende Tätigkeiten:
- *Ereignis "System aus " weitergeben,*
- *geforderte Daten an andere PDTen weitergeben,*
- *Daten sichern (z. B. Materialbestand, noch ausstehende Bestellungen),*

Nachbedingungen:
- *PDT in einem konsistenten Zustand,*
- *Daten über Zustand der Zelle gesichert.*

Beziehungen zwischen den Zustandswechseln von PDTen

Alle Tätigkeiten beziehen sich allgemein auf einen beliebigen PDT unter der Berücksichtigung der Kooperation mit anderen PDTs, damit das asynchrone Prozeßsystem FMZ seine Aufgaben erfüllen kann. Geht ein PDT vom Zustand 'funktionslos' in den Zustand 'arbeitet', so ist dieser Übergang nur sinnvoll, wenn alle anderen PDTen diesen Wechsel ebenfalls durchführen bzw. sich schon im Zustand 'arbeitet' befinden. Wechselt ein PDT in den Zustand 'wartet', weil z.B. eine Blockierung anderer PDTs oder ein Bedienereingriff vorliegt, so ist mit der Ausführung neuer Aktionen betroffener PDTen solange zu warten, bis der Eingriff beendet ist und alle PDTen mit aktuellen Daten weiterarbeiten können.

Diese Übergänge umfassen Tätigkeiten zum Sichern, Einlesen oder Aktualisieren von Daten, d.h. es handelt sich nach Voraussetzung um Übergänge, bei denen sich ein PDT in einem konsistenten Zustand befindet. Die Beseitigung der Unterbrechungsursache

kann auch Daten anderer PDTen ändern. Die Änderungen werden aber erst nach Abschluß der Behebung weitergegeben, um ein Weiterarbeiten anderer PDTen zu ermöglichen.

Aufwendiger ist der Übergang vom Zustand 'blockiert' in den Zustand 'arbeitet', denn hier befindet sich ein PDT in einem inkonsistenten Zustand, d. h. anhand der ermittelten Daten muß ein PDT zunächst in einen konsistenten Zustand gebracht werden, bevor der Normalbetrieb fortgesetzt werden kann. Dafür sind eine Vielzahl von Aktivitäten notwendig, die neben dem Ermitteln der Blockierungsursache und Korrigieren der fehlerhaften Daten, auch das Initiieren physischer Operationen in der Zelle umfassen.

Die Daten eines PDTs betreffen teilweise real vorhandene Objekte (z.B. Werkstücke, Transportmittel, Geräte). Demnach gestaltet sich die Wiederaufnahme der Montage nach einer Blockierung wesentlich schwieriger als in einem abgeschlossenen, abstrakten System. Neben der Behandlung physischer Vorgänge in der Zelle, müssen auch Fragen zur Synchronisation im vorliegenden Prozeßsystem berücksichtigt werden.

4.6.2 Wiederaufnahme nach geplanter Unterbrechung

Beim Wiederanlauf nach einer geplanten Unterbrechung, d.h. beim Übergang vom Zustand 'wartet' in den Zustand 'arbeitet' ist die Konsistenz der Daten eines PDTs genau dann gesichert, wenn die folgenden zwei Bedingungen erfüllt sind:

1. Die Daten eines PDTs müssen in sich konsistent sein.

Ausgehend von den Zustandsdefinitionen (Kap. 4.4) kann ein System von Gleichungen (Integritäts- oder Konsistenzbedingungen (Kap. 5.1)) aufgestellt werden, mit dessen Hilfe die logische Konsistenz der Daten überprüft wird. Mögliche Integritätsbedingungen können z.B. sein:

- Die Anzahl der Fertigungsaufträge (FA) ist gleich der Anzahl aller FAs im Zustand bekannt, begonnen oder beendet.
- Die Anzahl der FAs ist gleich der Anzahl der verschiedenen FA-nummern, die bei allen Teilaufträgen ermittelt werden.

2. Die Daten müssen das reale Abbild der Zelle beschreiben.

Um diese Forderung überprüfen zu können, muß das Prozeßsystem Einrichtungen zur Verfügung stellen, die Anzahl und Lage der real vorhandenen Objekte ermittelt und mit

den Daten des PDTs vergleichen kann (z.B Visualisierungssystem). Die Integritätsbedingungen und Funktionen zur Aufbereitung von Objektdaten sind auch notwendig, um eine automatische Ermittlung und Behebung von Unterbrechungsursachen durchzuführen.

Die weiteren Überlegungen aber werden unabhängig von Ausprägungen und Verfügbarkeit solcher Systemkomponenten (z.B. automatische Betriebsdatenerfassungs-, Auswerte- und Diagnosesysteme) durchgeführt. Ein PDT führt demnach zunächst solange seine Aktionen aus, bis er aufgrund des Eintretens eines Ereignisses (Pause, Bedienereingriff, Blockierung anderer PDTen) in den Zustand 'wartet' überwechselt.

Unterbrechung wegen Arbeitspausen

Bei Unterbrechungen wie Pause und Feierabend werden Zeitpunkte und Bedingungen (z.B. alle begonnenen AGs beenden) vorgegeben, die den PDT veranlassen, keine neuen Aktionen mehr auszuführen. Es können dann keine Änderungen mehr im System durchgeführt werden, die Daten des PDTs werden gesichert und nach Ablauf kann das System vom letzten Zustand vor der Unterbrechung wieder gestartet werden.

Unterbrechung wegen Eingriffe des Bedieners ins System

Diese Unterbrechung führt zwar auch in den Zustand 'wartet', bewirkt aber direkte Aktivitäten des Prozeβsystems. Bei der Wiederaufnahme des Normalbetriebs müssen diese Änderungen berücksichtigt werden, d.h die gesicherten Daten sind entsprechend zu aktualisieren und ein wartender PDT muβ geforderte Aktionen zur Gewährleistung der Konsistenz ausführen.

Da die Bezeichnung 'Bedienereingriff' eine Vielzahl von Tätigkeitsbereichen umfassen kann, seien hier einige Beispiele angeführt. Ein Bediener kann durch einen Eingriff ins System einen Transport oder einen Arbeitsgang eigenhändig ausführen bzw. auch anstoβen. Für die Behandlung dieser Tätigkeiten sind Kenntnisse (z.B. durch BDE und Visualisierung) wichtig, ob der aktuelle Transport oder Arbeitsgang, ein bereits eingeplanter und zur Ausführung anstehender oder ein beliebiger Transport oder Arbeitsgang vom Bediener ausgeführt wurde. Im letzten Fall müssen nur die Änderungen am Material oder den Betriebsmitteln berücksichtigt werden, während in den zuvor genannten Fällen auch die ausgeführten Arbeitsgänge als 'abgeschlossen' zu kennzeichnen sind. Ebenso kann ein Bediener Werkstücke oder Betriebsmittel aus der Zelle entfernen oder zusätzliche Elemente in die Zelle bringen.

Unterbrechung durch Blockierung anderer PDTen

Es kann notwendig sein, daß bei der Blockierung eines PDTs alle anderen PDTen in den Wartezustand übergehen, bis die Blockierungsursache beseitigt wurde. Ist die Blockierung behoben, so kann ein wartender PDT wieder in den Zustand 'arbeitet' übergehen, indem seine gesicherten Daten aktualisiert und geforderte Aktionen vor Aufnahme des Normalbetriebs durchgeführt werden.

Nach einer Unterbrechung durch Bedienereingriffe und durch Blockierung anderer PDTen kann es sinnvoll sein, Tests durchzuführen, um z. B. die Konsistenz der Daten zu überprüfen und sicherzustellen, daß bei einem Bedienereingriff alle Änderungen angegeben wurden, oder um zu prüfen, ob alle anstehenden Anforderungen noch ausgeführt werden sollen.

Hierzu seien noch einige Anmerkungen zur unterschiedlichen Verwendung der verschiedenen Datenklassen zur Erreichung eines konsistenten Zustandes angeführt. So werden z.B. Sollvorgaben aus übergeordneten Planungsinstanzen und Stammdaten der Zelle während des Montagevorganges nicht verändert, d.h. ihre Werte können jederzeit aus einer Datenbasis erneut übernommen werden. Bei Unterbrechungen erfordern sie daher keine Behandlung in Form von Sichern oder Aktualisieren.

Intern generierte Steuerdaten und Rückmeldungen/Quittungen, werden unabhängig einer späteren Verwendung im Normalbetrieb gesichert. Denn sie bilden die Grundlage zur Unterbrechungsbehandlung, zur Fehlerlokalisation und Wirkungsforschung. Zustandsdaten dagegen werden für die Unterbrechungsbehandlung und den Wiederbeginn der Montage benötigt. Physische Änderungen müssen in der Datenbasis berücksichtigt werden, d.h. es erfolgt eine Anpassung an das tatsächliche Abbild.

Insgesamt sollte weiterhin für den Aufbau eines Systems beachtet werden, daß alle Funktionen, die für den Übergang 'wartet -> arbeitet' gelten, auch uneingeschränkt für den Zustandswechsel 'blockiert -> arbeitet' verwendet werden können, sobald eine konsistente Datenbasis geschaffen ist.

4.6.3 Wiederaufnahme nach Blockierung des Prozeßsystems

Im vorliegenden Kapitel werden die grundsätzlichen Methoden für das Wiederherstellen eines konsistenten Zustands vorgestellt. Zugleich wird die praktische Anwendbarkeit für die Steuerung einer flexiblen Montagezelle untersucht und die Probleme angesprochen, die sich aufgrund eines komplexen Prozeßsystem (Vielzahl von kooperierenden PDTen) und der Berücksichtigung physischer Operationen in der Zelle ergeben. Grundsätzlich werden in der Literatur die zwei Methoden Recovery und Reset zum Wiederherstellen eines konsistenten Zustandes unterschieden /86/. Bei beiden Methoden können abhängig von der Zielrichtung der durchgeführten Tätigkeiten nach Forward- oder Backwardmaßnahmen unterschieden werden.

Recovery

Bei einer Recovery wird vom inkonsistenten Zustand aus versucht, schrittweise einen konsistenten Zielzustand zu erreichen. Als Zielzustand bieten sich besonders an:

- der letzte konsistente Zustand vor der Blockierung oder
- der erwartete konsistente Zustand, der bei einer fehlerfreien Ausführung erreicht worden wäre.

Backwardrecovery:

Sie umfaßt Maßnahmen zum Rückgängigmachen der Effekte von Aktionen, die seit Auftreten des Fehlers ausgeführt wurden, d.h. dabei soll ein davorliegender konsistenter Zustand erreicht werden. Die Durchführung von Maßnahmen zur Backwardrecovery setzt vorraus, daß

- alle Daten der bereits erreichten Zustände bekannt oder die Daten des Anfangszustandes und alle durchgeführten Aktionen verfügbar bzw. ermittelbar sind,
- für die durchgeführten Aktionen, deren Effekte rückgängig gemacht werden sollen, entsprechende inverse Aktionen zur Verfügung stehen,
- die Fehlerursache ermittelt und beseitigt werden kann, damit bei Wiederholung keine erneute Unterbrechung eintritt.

Forwardrecovery:

Kennzeichen der Forwardrecovery sind Maßnahmen, die vom inkonsistenten Zustand aus den erwarteten konistenten Zustand oder einen anderen zukünftigen Zustand erreichen. Um sie durchführen zu können, muß es daher möglich sein

- die Daten des erwarteten Zustandes oder eines anderen zukünftigen Zustandes zu ermitteln,
- Aktionen zum Erreichen des Zielzustandes zu generieren.

Reset

Nach Auftreten einer Unterbrechung wird die Bearbeitung an einem zuvor festgelegten Wiederaufsetzpunkt, der auch als Sicherungspunkt bezeichnet wird, fortgesetzt. Ein Sicherungspunkt umfaßt dabei alle Informationen zum Wiederherstellen eines konsistenten Zustandes. In jedem System kann somit der Anfangszustand auch als ausgezeichneter Sicherungspunkt angesehen werden. Auch für das Festlegen von Sicherungspunkten gibt es eine Reihe von Möglichkeiten:

- Im Laufe der Bearbeitung werden Sicherungspunkte erstellt. Im Fall einer Unterbrechung wird das System auf den aktuellsten Sicherungspunkt zurückgesetzt,
- Für jeden Bearbeitungsabschnitt wird ein Sicherungspunkt festgelegt, bei dem im Blockierungsfall aufgesetzt wird.

Für die Festlegung von Sicherungspunkten wirken sich folgende Punkte auf den Aufwand durchzuführender Maßnahmen aus (vgl. Bild 32):

- Umfang des Sicherungspunktes
  (z.B. alle Daten des Systems, nur Änderungen)
- Zeitintervall
  (z.B. pro Schicht, pro Stunde)
- Erstellungsvorgang
  (z.B. direkt nach Ablauf einer Zeitspanne, Einreihung
  in andere anstehende Aktivitäten)

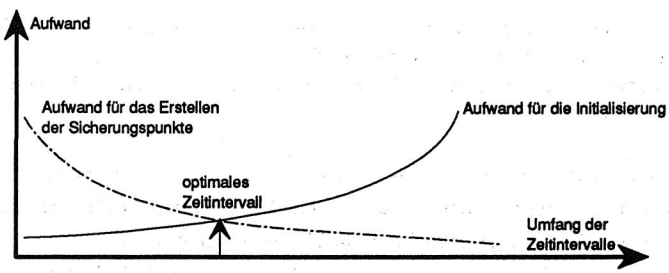


Bild 32: Aufwandsabschätzung für Initialisierungsmechanismen

Im Zusammenhang mit der Erreichung eines konsistenten Zustandes eines PDTs sei das Problem des 'Dominoeffektes' erwähnt. Erfordert ein PDT ein Zurücksetzen seiner Aktionen, so kann es notwendig werden, andere PDTen ebenfalls zurückzusetzen, da

diese Ergebnisse des ersten PDTs verwendet haben. Dies kann wiederum ein Zurücksetzen des ersten PDTs erfordern usw., bis sich die PDTen wieder im Anfangszustand befinden. Dieses Verhalten asynchroner PDTen kann unter bestimmten Voraussetzungen umgangen werden. Eine Möglichkeit besteht darin, daß alle PDTen ihre Sicherungspunkte zum selben Zeitpunkt erstellen, weitere Strategien sind in /102/ beschrieben. Für weitere Betrachtungen wird von dominoeffektfreien PDTen ausgegangen.

Recovery- und Resetmaßnahmen in der Steuerung

Recovery- bzw. Resetmaßnahmen erfordern Zeit und Kosten, sie sind daher nur sinnvoll, wenn der Aufwand, den sie verursachen, geringer ist als der Aufwand, der durch den Neustart eines Systems vom Anfangszustand aus verursacht wird. Da sich in der Steuerung einer FMZ lange Unterbrechungszeiten oder sogar ein Neustart vom Anfangszustand aus mit den Forderungen nach maximaler Flexibilität nicht vereinbaren lassen, sind Recovery- bzw. Resetmaßnahmen unumgänglich.

Die Wiederaufnahme der Montage nach einer Unterbrechung erfordert die im folgenden zusammengefaßten Daten, Funktionen und Methoden:

- Eine redundante Datenhaltung, d.h. alle Daten des Prozeßsystems sind zusätzlich zu den lokalen Datenbereichen ein weiteres Mal (z.B. in einer Datenbank) abgespeichert. Nach einer festzulegenden Zeitspanne werden alle Änderungen in diese Datenbank übertragen, auf die bei Verlust oder Störung zurückgegriffen werden kann.
- Kontrollinformationen, dazu zählen Informationen über ausgeführte Aktionen sowie Meldungen oder Quittungen.
- Ersatzaktionen zu allen Aktionen eines Prozeßsystems. Das sind Aktionen, die Auswirkungen beendeter oder abgebrochener Aktionen rückgängig machen.
- Funktionen zum Erkennen und Beheben von Fehlerursachen, die für eine Unterbrechung im Fehlerfall benötigt werden.

Das Zurücksetzen von Aktionen ist für abstrakte Prozeßsysteme immer möglich (vgl. Recoverymaßnahmen in Datenbanken /86/) und im allgemeinen einfacher als Forwardmaßnahmen, da der konsistente Zielzustand bereits bekannt ist.
Die abstrakten Aktionen des asynchronen Prozeßsystems FMZ können jedoch nicht losgelöst von den dazugehörigen physischen Vorgängen in der Zelle betrachtet werden. Wird nämlich eine 'abstrakte Aktion' eines PDTs rückgängig gemacht, so müssen auch evt. dazugehörige Tätigkeiten zurückgenommen werden. Da physische Vorgänge nicht immer reversibel sind, können Maßnahmen zur Backwardrecovery oder ein Reset nur unter bestimmten Voraussetzungen angewendet werden. Aus diesen Gründen sind Methoden zur Recovery oder zum Reset in flexiblen Montagezellen schwieriger und umfangreicher als in abgeschlossenen abstrakten Prozeßsystemen.

Unter bestimmten Voraussetungen anwendbar heißt hier:
- Aktionen eines PDT, die keine physischen Operationen der Montagezelle initiieren, können immer zurückgesetzt werden, z. B. das Reservieren von Betriebsmitteln.
- Für viele Operationen, zu denen physische Operationen gehören, existieren ebenfalls inverse Operationen, z. B. für eine Transportaktion.
- Für Operationen mit keiner existenten inversen Operationen gibt es eine Reihe von Ersatzoperationen.

Zusammenfassend läßt sich feststellen, daß Rückwärtsverfahren nur anwendbar sind, wenn von den vorhandenen physischen Vorgängen entschieden werden kann, ob sie reversibel sind bzw. inwieweit Operationen zur Beseitigung unerwünschter Auswirkungen zur Verfügung stehen. Um genaue Aussagen darüber machen zu können, müssen notwendige Tätigkeiten auch in Abhängigkeit der Anwendung, d.h. von konkreten PDTen (vgl. AV, MDP, ... in Kap. 7), analysiert werden.

Für Untersuchungen von Recovery- und Resetmaßnahmen ist es notwendig für asynchrone Prozeßsysteme bzw. PDTen die zu betrachtende Einheit festzulegen. So wird bei Datenbanksystemen als Einheit für Recoverymaßnahmen die 'Transaktion' verwendet /86/, nach deren korrekter Ausführung die Datenbank wieder in einem konsistenten Zustand ist.

Eine Aktion (Teilaktion) - vergl. Modell in 4.1.2 -, die durch eine Nichtblockierungsbedingung und Effektbeschreibung festgelegt ist, überführt bei fehlerfreier Ausführung einen abstrakten synchronisierten PDT von einem konsistenten Zustand wieder in einen konsistenten Zustand. Dabei ist der Umfang einer Aktion von der Schicht abhängig, in der sie verwendet wird. In Schicht 4 umfaßt eine Aktion z.B. die Ausführung eines Arbeitsschrittes, während in Schicht 5 eine Aktion z.B. aus der Zuteilung von Ressourcen für einen Arbeitsgang besteht. Eine Aktion wird nur genau dann fehlerfrei ausgeführt, wenn sie mit fehlerfreien Eingabedaten ohne Unterbrechung zu Ende bearbeitet wird. Eine Aktion ist genau dann fehlerhaft ausgeführt, d.h. sie überführt einen PDT in einen inkonsistenten Zustand, wenn sie

- ohne Unterbrechung ausgeführt wird, die Eingabedaten jedoch fehlerhaft sind,
- fehlerfreie Eingabedaten vorliegen, die Bearbeitung aber vor dem definierten Ende unter- bzw. abgebrochen wird,
- falls beide Situationen gleichzeitig eintreten.

Der bei /52/ beschriebene Lebenszyklus einer Aktion ist im Hinblick auf Backward- oder Forwardmaßnahmen von Bedeutung. Danach wird der Zyklus, wie in Bild 33 dargestellt, in drei Abschnitte eingeteilt.

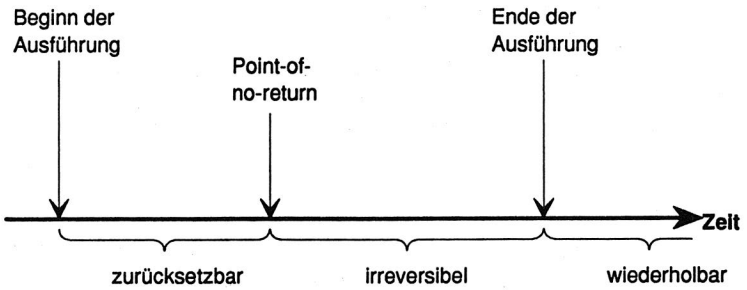


Bild 33: Lebenszyklus einer Aktion

Initiiert eine Aktion eines PDTs auch eine physische Operation, dann gilt: soweit noch keine physische Operation angestoßen wurde, befindet sich die Aktion im Zustand 'zurücksetzbar'. Wurden bereits physische Operationen begonnen, so befindet sich die Aktion im zweiten Abschnitt; sie hat damit den sog. Point-of-no-return überschritten. Nach Ende der Ausführung hat sie den dritten Lebensabschnitt, den Zustand 'wiederholbar', erreicht /52/.

Für die Behandlung unterbrochener Aktionen ist der Zeitpunkt der Unterbrechung wichtig. Tritt eine Unterbrechung im ersten Abschnitt auf, so sind nur abstrakte Aktionen zu behandeln, für die Rücksetzmaßnahmen immer möglich sind. Bei einer Unterbrechung im zweiten Abschnitt müssen neben Maßnahmen zur Recovery und zum Reset von abstrakten Aktionen auch die bis dahin erzielten Auswirkungen physischer Operationen behandelt werden, so daß in Abhängigkeit davon Forwardmaßnahmen durchgeführt werden.

Eine weitere Möglichkeit besteht darin, daß eine Unterbrechung zu dem Zeitpunkt auftritt, wo eine physische Aktion bereits ausgeführt wurde, die korrekte Ausführung aber noch nicht durch Änderung der Daten dem Prozeßsystem sichtbar gemacht wurde. In diesem Fall ist nur eine entsprechende 'Nachbehandlung' der betroffenen Daten erforderlich.

4.7 Anbindung an übergeordnete Systeme

Die entwickelten Mechanismen und daraus abgeleitete Abläufe in einer Zelle können jedoch nicht isoliert betrachtet und eingesetzt werden. Von groβer Bedeutung ist die informationstechnische Verknüpfung mit den übrigen Komponenten der Produktionsausführung, d.h. die Bereitstellung notwendiger Eingangsdaten durch die Auftrags- und Montageplanung sowie die Entgegennahme generierter Rückmeldungen. Die Anforderungen betreffen vor allem die Rahmenarbeitsplan-, Auftrags- und die Stammdaten der Zelle.

Diese Planungs-Komponenten müssen im zeitlichen Vorfeld der Montagesteuerung aktiv sein und die notwendigen Voraussetzungen zum Ausschöpfen der Intelligenz vor Ort im Zellensystem erbringen. Die Auftragsplanung hat z.B. die Produktionsauslastung abzustimmen, denn sowohl eine kapazitiv überlastete als auch eine durch zu enge Liefertermine in Zugzwang gebrachte Montage läβt sich nicht mehr effektiv, also agierend steuern, da jeglicher Steuerungsspielraum fehlt /58/, sodaβ nur bindende Vorgaben ausgeführt werden. Eine Auftragsleitstelle kann in diesem Umfeld dann als Instanz für folgende Aufgaben eingesetzt werden:

- Einlasten von Aufträgen in die Zelle,
- Eckterminüberwachung, Aufzeigen absehbarer Terminverzüge,
- die Verfolgung von Teilen, Bestellempfänger der Zelle und Befehlsgeber von Transportsystemen (z.B. FTS),
- die Dokumentation von Rückmeldungen zur Aufbereitung neuer Auftragsunterlagen.

Zu einem speziellen Auftrag müssen alle notwendigen Informationen (z.B. eine auftragsorientierte Erzeugnisstruktur) geliefert werden. Die dem Auftrag zugeordneten Montageteile müssen in den Teilestammdaten vollständig beschrieben sein, ansonsten ist eine Ergänzung bei Übergabe des Auftrages notwendig (durch Aufforderung oder Kenntnis der Arbeitsvorbereitung).

Die Montageplanung ist u.a. für die Erstellung des Montagenetzplanes und des Ablaufplanes zuständig, wozu auch Aufgaben gehören wie:

- die Ermittlung von Vorgabezeiten für neue zellen- bzw. stationsspezifische Montageoperationen,
- das Ableiten realistischer Montageablaufpläne aus dem Netzplan als Gesamtgraph für eine Zelle.

Sowohl für das Ermitteln der Vorgabezeiten (wichtig für die Belastung der Zelle und einzelner Stationen) als auch für die Ermittlung der Gesamtmontagezeiten existieren eine Reihe zuverlässiger Verfahren (z.B auf der Grundlage der Netzpläne das PERT-Verfahren -Program Evaluation and Review Technique), die nicht nur exakte Vorgabezeiten, sondern auch streuende Erfahrungswerte verarbeiten können /10/.

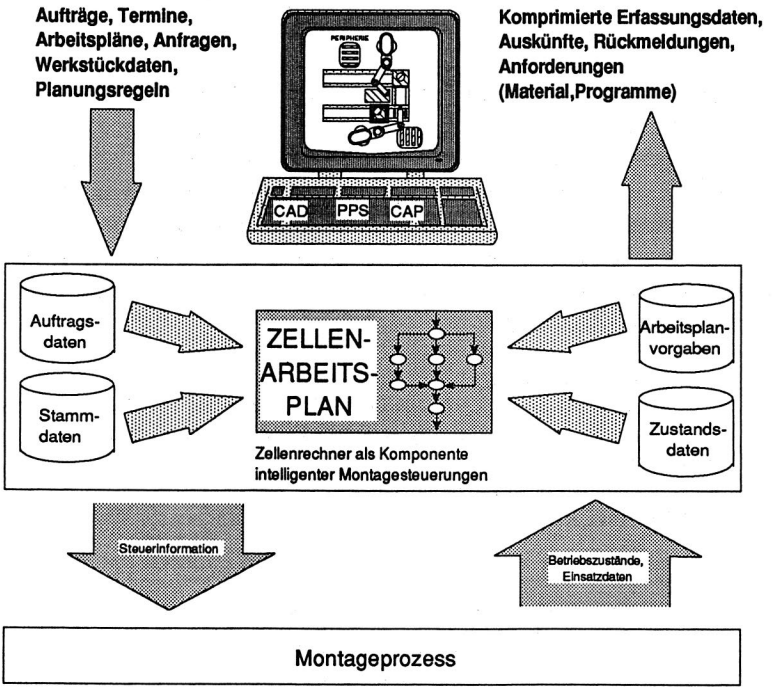


Bild 34: Vorgaben für die Montagezelle aus vorgelagerten Bereichen

Realistische Montageablaufpläne müssen neben der Planmontagezeit einen störungsbedingten Durchlaufzeitanteil enthalten. Ein solcher Faktor läßt sich mittels bekannter stochastischer Methoden bestimmen /6/. Das Ergebnis sind quantifizierte Ablaufpläne, die eine realistische Eingangsgröße für die Montagesteuerung bilden.

Durch Regeln oder im Dialog sind mehrere alternative Ablaufpläne zu reduzieren. Jeweils frei wählbare, aber dann feste Algoritmen und Regeln sind vorzuziehen, da sie die Reproduzierbarkeit der Ablaufpläne für spätere Auswertungen erhöhen, einem Bediener Eingriffsmöglichkeiten bieten und im Störungsfalle sowohl manuell als auch automatisch schnellere Reaktionen erlauben.

In bezug auf den Arbeitsplan hat die Planungsebene außerdem insbesondere AG-Daten und Informationen zum Parametrisieren der zugehörigen Arbeitsschritte bereitzustellen, z.B. über genormte Schnittstellen (vgl. Kap. 2.3.3). Daten zur Benennung der Teile und die Inputdaten müssen nur klassenmäßig mit Verweisen auf Knoten einer Produktstruktur (z.B. aus CAD Entwurf ableitbar) bereitgestellt werden.

Die globale Produktstruktur für eine Zelle ist also nur bei Einführung einer neuen Produktfamilie zu entwerfen bzw. bei Änderungen des Produktes oder der Ressourcen abzuwandeln. Weiterhin muß sichergestellt sein, daß je nach Intelligenz der Zellensoftware die zur Ausführung nötigen Daten (z.B. Umweltmodell oder RC-Programme) vorhanden sind (Bild 34).

Die Stammdaten werden entweder bei der Konfiguration oder bei Einführung weiterer Zellenelemente bzw. Ressourcen (Paletten, Magazine, Werkzeuge, Teile etc.) in den Datenbestand aufgenommen. Anzustreben ist hierbei, daß über das Betriebsmittelmodell und Produktmodell aus der Konstruktion und Arbeitsvorbereitung die notwendigen Gerätedaten, Daten über den Peripherieaufbau und Geometriedaten (Fügeorte, Fügeendlage,...) zu den montierbaren Teilen (z.B. über gemeinsame Datenbank) zur Verfügung gestellt werden.

5 Mechanismen zur Unterstützung bei der Realisierung

5.1 Relationale Datenbanken

Bei technischen Anwendungen kann für verschiedene Anwendungsprozesse der jeweilige Funktionsumfang meist von vorne herein vorgegeben werden. Damit ist es auch möglich, jedem verarbeitenden Prozeβ einen individuellen Datenraum zuzuordnen. Hierbei kann die physische Verteilung des allgemeinen Datenbestands durch das Need-to-know-Prinzip unterstützt werden /52/.

Für Anwendungen in der Montagesteuerung bedeutet dies, daβ die benötigten Daten aus anderen Teilbereichen, wie zum Beispiel Geometriedaten aus der Konstruktion oder Strukturgraphen aus der Arbeitsvorbereitung, auch direkt über die Datenbank zur Verfügung stehen können (Bild 35). Zudem kann eine Aufschlüsselung in die Datenbereiche der verschiedenen Abstraktionsstufen der Zellensteuerung schon auf der Schemaebene eines Datenmodells vorgenommen werden. Die Auflistung der Abhängigkeiten (z.B. durch Relationen) zwischen Daten der einzelnen Schichten kann letztendlich auch mit entscheidenden Performanceverbesserungen verbunden sein.

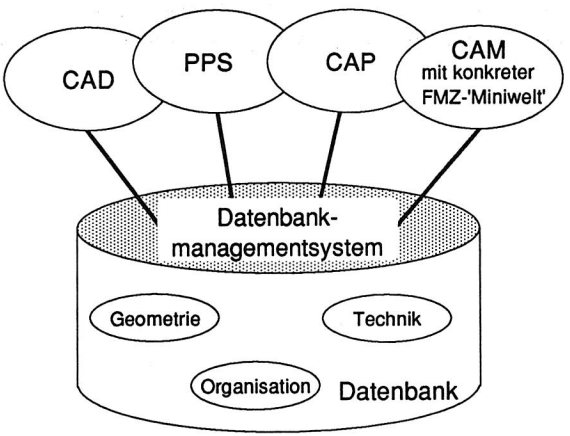


Bild 35: Datenbank aus der Sicht einer FMZ

5.1.1 Prozeßnahe Anwendung

Bei den folgenden Überlegungen steht die spezielle Problematik der lokalen Datenhaltung auf prozeβnahen Anwendungsebenen im Vordergrund. Neben den damit verbundenen Nichtstandardaufgaben für Datenbanken wie relationale Geometriemodelle oder Langzeittransaktionen interessieren in diesem Zusammenhang besonders die Anforderungen in bezug auf Realzeitverhalten. Allgemein wird für Realzeitsysteme gefordert, daβ ein maximales Zeitintervall dt_{max} existiert, das zwischen der Wahrnehmung einer Zustandsänderung und der Systemreaktion nicht überschritten werden darf.

Neben den verschiedenen Lösungsansätzen auf Seite der Anwendungsprozesse, wie spezielle Prozeβstrukturen und Fehlertolerierungsmechanismen, müssen auch die verwendeten Datenhaltungssysteme die Anforderungen von Realzeitanwendungen berücksichtigen. Hierzu sind Verfahren bereitzustellen, mit denen auf die Reaktionszeiten bei konkurrierendem Zugriff Einfluβ genommen werden kann, so z.B. durch

- Prioritätenvergabe: Die Transaktionsprogramme aus den Anwendungsfunktionen erhalten hier entsprechend ihrer Bedeutung als Realzeitoperation Abarbeitungsprioritäten zugewiesen.
- Sychronisationsverfahren: Üblichen Sperrverfahren zur Zugriffssynchronisation auf die DB-Objekte sind bei Realzeitanforderungen etwa Multiversionskonzepte vorzuziehen.
- Compilierung der Transaktionsprogramme:Im Vergleich zur Interpretierung der Zugriffsbefehle ermöglicht eine Präcompilierung kürzere Zugriffszeiten.
- Dynamische Zugriffspfadoptimierung: Mit Hilfe von Informationen aus einem Data Dictionary und dem vorher analysierten Schema der Zugriffe einer Transaktion ist die Optimierung der Zugriffspfade möglich.
- Clustern von Datensätzen: Logisch verwandte Datenmengen sollten zwecks Zugriffszeitverkürzung sowohl physisch auf dem Datenmedium als auch logisch über die vorhandenen Zugriffsstrukturen geclustert, d.h. zusammenhängend gespeichert werden.
- Prefetching: Im Gegensatz zu üblichen Demand-Paging-Verfahren werden Datenseiten bereits vor der Anforderung eingelesen und im DB-Puffer gehalten.

5.1.2 Relationale Datenmodelle zur Strukturierung

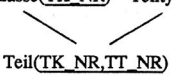
Grundlagen bezüglich des Einsatzes relationaler Datenmodelle betreffen neben der Entwurfssystematik und der verwendeten Darstellungsform auch die Bedeutung der semantischen Elemente, d.h. die Integritätsbedingungen und Operatoren /43, 52, 105/.

Entwurfssystematik

Bei der Entwicklung konzeptioneller Datenmodelle existieren zwei grundlegende Vorgehensweisen. Neben der Top-Down-Methode, bei der zuerst die zu modellierenden Begriffe definiert werden und diese durch Normalisierung in eine brauchbare Form gebracht werden, wird häufig der umgekehrte Weg verfolgt. Bei der Bottom-Up-Methode (domainorientierter DB-Entwurf) werden zuerst über die notwendigen Wertebereiche die zugeordneten Attribute bestimmt. Aus diesen Attributen bildet man dann die Basisrelationen des Datenmodells. Im weiteren Verlauf des Entwurfsprozesses können diese Basisrelationen dann zu komplexeren Objekten zusammengesetzt werden. Dazu gibt es drei Möglichkeiten:

Komposition: Aus zwei einfacheren Begriffen entsteht durch Zusammensetzen der elementaren Schlüssel ein komplexer.

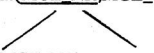
Beispiel: Teilklasse(TK_NR) Teiltyp(TT_NR)



Teil(TK_NR,TT_NR)

Subordination: Einfache Begriffe werden unter einem Oberbegriff zusammengefaßt. Der Wertebereich der Primärschlüsselattribute bleibt erhalten. Der Oberbegriff erhält zusätzlich ein diskriminierendes Attribut.

Beispiel: Magazin(MGZ_NR,MGZ_TYP)



Teilemagazin(MGZ_NR) Werkzeugmagazin(MGZ_NR)

Reduktion: Die Reduktion realisiert die Abstraktion von einem komplexen Begriff auf einen einfacheren. Dieser Effekt kann auch umgekehrt mittels Komposition unter Zuhilfenahme von Integritätsbedingungen realisiert werden.

Beispiel: Magazininhalt(MGZ_NR,TEIL_NR)

Magazin(MGZ_NR)

Darstellungsform

Die Elemente des Datenmodells werden als sogenanntes Entity-Relationship-Diagramm dargestellt. Neben der Entstehung der komplexen Begriffe aus den einfachen spiegeln diese Diagramme auch die Schlüsselbeziehungen zwischen den Relationen wider. Die aus den Subsumptionen entstehenden komplexen Objekte heißen nach ihrer Entstehungsweise Komposition, Subordination und Reduktion.

Integritätsbedingungen (Daten- und operationale Integrität):

Allgemein werden Integritätsbedingungen als Regeln für die Vergabe von Attributwerten in einem Datenbankmodell angesehen. Einfeldbedingungen beziehen sich nur auf den Wert eines Attributs.

Beispiel: Ein einzutragendes Datum muß für ein bestimmtes Attribut in der Zukunft liegen.

Mehrfeldbedingungen drücken in einem Relationenschema die Abhängigkeiten zwischen Attributen einer Relation aus.

Beispiel: Auftrag(FA_NR,Anz_zu_fertigen,Anz_gefertigt)
Anz_gefertigt <= Anz_zu_fertigen

Mehrfeldbedingungen können aber auch über mehrere Relationenschemata, die Abhängigkeiten zwischen Attributen verschiedener Relationen betreffen, eingesetzt werden.

Beispiel: Auftrag(FA_NR,Status, ...)
Teilauftrag(TA_NR,FA_NR, ...)
Zu jeder FA_NR x in Teilauftrag existiert ein
Tupel in Auftrag mit FA_NR=x und Status=zerlegt

Daneben lassen sich Integritätsbedingungen auch in konstitutive und regulative Integritätsbedingungen einteilen:

konstitutiv: Attributwerte aus zusammengesetzten Schlüsseln oder Fremdschlüsseln sind nur gültig, wenn zu diesen Werten entsprechende Tupel in den Ausgangsrelationen existieren.

Beispiel: Teil(TK_NR,TT_NR)
Teilklasse(TK_NR)
Zu jedem Teil muß ein Eintrag in Teilklasse
mit der entsprechenden TK_NR existieren.

regulativ: Integritätsbedingungen als 'semantische Regeln'

Beispiel: Rahmenarbeitsplan(RAPL_NR,Produktfamilie)
Für jede Produktfamilie existiert nur ein RAPL.

Operatoren

Die Operatoren zur Datendefinition, -manipulation und -kontrolle auf die Objekte einer Datenbank sind Bestandteil der Semantik des modellierten Anwendungsbereichs. Sie unterstützen das System bei der Einhaltung der geforderten Integritätsbedingungen und

regeln den geordneten Zugriff auf die Elemente des Modells. Über die Verwendung des Transaktionskonzepts in den Operatorfunktionen kann die Konsistenz der Daten gewährleistet werden.

5.2 Einsatz und Vorteile von UNIX Systemen

Marktanalysen zufolge wird sich das Dickicht von verschiedenen Betriebssystemen zukünftig stark lichten. Neben den Betriebssystemen MVS und VM in der IBM Welt und VMS bei Digital Equipment werden namhafte Hersteller meist auf das Betriebssystem UNIX setzen, das hardwareunabhängig ist und somit für Integrationszwecke - wie es in der Produktion gefordert wird - prädestiniert erscheint.

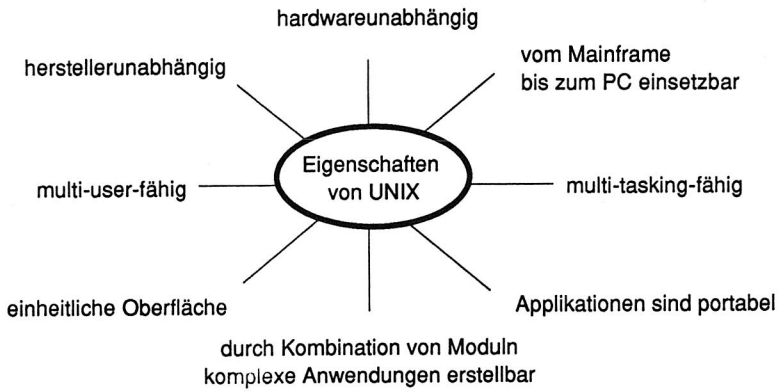


Bild 36: Eigenschaften von UNIX Systemen

Im Bereich der Personal Computer wird sich aufgrund der großen Marktdurchdringung besonders im Verwaltungsbereich MS-DOS behaupten /11/. UNIX hat aber den Vorteil, das es universell auf allen Rechnerebenen einsetzbar ist und somit für die dezentrale Datenverarbeitung verfügbar ist. Dies trägt nicht zuletzt zur Bildung eines 'offenen Systems' mit horizontalen und vertikalen Erweiterungsmöglichkeiten bei, so daß jeweils eine Durchgängigkeit vom dispositiven Bereich bis zur operativen Ebene (Anbindung unterschiedlicher elementarer Steuerungen) erreicht wird. Die Eigenschaften von UNIX in bezug auf eine mögliche Integration von Anwendungssoftware in den Produktionsbereich sind in Bild 36 zusammengefaßt.

5.2.1 Aufbau der Systemsoftware

Für kein anderes Betriebssystem werden Standardisierungsentwicklungen durch Hersteller, Benutzergruppen und neutrale Normierungsgremien so forciert wie für UNIX (z.B. X/OPEN Group). Die Aufgabe ist die Definition und Durchsetzung einer standardisierten Anwendungsumgebung, die portable Anwendungssoftware und Integration sicherstellt. Bei den einzelnen Komponenten handelt es sich dabei um Sprachen, das Datenmanagement, Netzwerke, Benutzer-Schnittstellen, also um standardisierte Schnittstellen, die eine Integration der herstellerspezifischen Software ermöglichen.

UNIX ist heute bereits Weltstandard bei herstellerunabhängigen Betriebssystemen. Der Begriff "UNIX System V" steht in der Fachwelt für umfassende Portabilität zwischen den verschiedensten Prozessoren und Rechnern. Für den Anwender wird dies zum Vorteil, wenn er auf dem Markt für seine Anwendung Softwarekomponenten vorfindet.

Im Vergleich zu anderen Betriebssystemen ist UNIX mit seinen Hilfsprogrammen gut für die Softwareerstellung geeignet. Weitere Aspekte für den Einsatz in der Fertigung betreffen den Hintergrundbetrieb (z.B. für Überwachungs- und Protokollierungsaufgaben), die abgestufte Zugangskontrolle (Systemmanager - Bedienpersonal), den abgestuften Daten- und Dateischutz (z.B. wichtig in bezug auf Stammdatenänderung), die Unterstützung gemeinsamer Betriebsmittelnutzung (Ausgabemedien) und geeigneter Fenstertechniken (Bedienermenüs). Hinzu kommt das Angebot der Bibliotheken (z.B. die Standards zur Ein-/Ausgabe) sowie der internen Kommunikationsroutinen.

5.2.2 Echtzeit - Aufgaben

Für Echtzeitaufgaben ist das Betriebssystem UNIX in seiner Standardimplementierung aus unterschiedlichen Gründen (Reaktionszeit, Datendurchsatz) nicht geeignet. Aber gerade in der Fertigung kommt im Vergleich zur kommerziellen Datenverarbeitung dieser Aspekt zum Tragen, wenn ein Prozeß zu bearbeiten ist, der eine bestimmte Reaktion auf asynchrone äußere Ereignisse (z.B. Fehlermeldung einer Maschinensteuerung) innerhalb einer vorgesehenen Zeitspanne erfordert. Bei den meisten Anwendungen im Echtzeitbereich sind viele externe asynchrone Ereignisse zu verarbeiten, von denen oft jedes andere Reaktionen und Reaktionszeiten erfordert und die dazu noch von unterschiedlicher Wichtigkeit sind. Durch mögliche Modifikationen zum Anschluß an Echtzeitsysteme bzw. durch Erweiterungen zu Realtime kann UNIX und UNIX-Derivate als geeignetes Betriebs- und Entwicklungssystem für Zellenrechner angesehen werden.

In bezug auf die Erfüllung dieser komplexen Anforderungen werden unter UNIX Systemen zwei Wege beschritten:

- Erweiterung des UNIX Kerns mit Echtzeitfunktionen
- Integration einer eigenständigen Kommunikationsbaugruppe

Zu den Eigenschaften einer Erweiterung zählen neben zentralspeicherresidenten Prozessen eine Prozeβverwaltung mit verdrängenden Prozessen (sog. Preemptive Scheduling) und Prioritätsstufen für Interrupt-Routinen. Die Auslagerung zeitkritischer Aufgaben auf ein Schnittstellensystem entlastet insgesamt den Rechner zur Erfüllung von Verwaltungs- und Koordinierungsaufgaben und fördert zugleich die Schaffung einer geräteunabhängigen Schnittstelle. Eine Kommunikationsbaugruppe mit eigenem Prozessor stellt weiterhin meist weitere Schnittstellen zur Verfügung und erlaubt somit die Kommunikation mit mehreren Gerätesteuerungen sowie ein paralleles Fahren unterschiedlicher Schnittstellenprotokolle (unterschiedliche Interpreter und Treibermodule).

In diesem Rahmen kann auf Weiterentwicklungen in Verbindung mit der Ebene 3 , die einen Echtzeitkern bei Hybridsystemen /100/ betreffen, verwiesen werden. Die Möglichkeit der Einbindung von extern erstellten C - Programmen unterstützt auch die geforderte Durchgängigkeit von der dispositiven Zellenebene zu der Geräteebene. Die Integration erfolgt hier über den PDV Bus, der als einheitliches Bindeglied zu allen Einzelgeräten dient. Somit ist der Anschluβ der gesamten Sensorik - auch der Einfachsensoren - über das Kommunikationssystem gewährleistet, das verschieden mächtige Kommunikationspartner unterstützt.

5.3 Strukturen von Kommunikationsmedien

Für topologische und organisatorische Strukturen von Kommunikationsmedien auf der Fertigungsleitebene haben die weltweiten MAP-Aktivitäten bereits zu einer Palette kommerziell verfügbarer Produkte auf Basis genormter lokaler Netzwerke geführt. Für die Feldebene dagegen, die durch strenge Zeitgrenzen und durch das Zusammenspiel komplexer und einfacher Geräte (z.B. Robotersteuerungen versus Identifikationssysteme) gekennzeichnet ist, läβt die MAP Protokollhierarchie keine leistungsfähigen und technisch sinnvollen Lösungen zu.

So sind zwar Aktivitäten zur Schaffung einer internationalen MAP konformen Feldbus-Norm (PROFIBUS) im Gange /8/, aber beim Stand der Technik verbleibt noch die Problemstellung in der Montageautomatisierung, Steuerungsgeräte auch über serielle

Verbindungen (RS232, V24) technisch zu koppeln. Als Industriestandard können aufgrund einer umfangreichen Produktpalette die auf der RS-485 Norm gründenden BITBUS-Konfigurationen (Single Master-Bus) /15/ angesehen werden. Innerhalb der Normungswelt sollte in Zusammenhang mit MAP für neue Anwendungen auf die Aktivitäten von MMS und auf die jeweiligen Companion Standards geachtet werden, hier also auf die RC-CS und NC-CS, die sich speziell auch mit Kommunikationsanforderungen in Montagezellen beschäftigen /5, 16/.

5.4 Verwendung der Softwaretechnologie

Der wirtschaftliche Zwang, das in der Wiederverwendbarkeit schlummernde Potential auszunutzen, verstärkt sich immer mehr. Es erscheint wichtig, daß man Erfahrungen systematisch auswertet, auf den Erfolgen aufbaut und die Gründe für die Mißerfolge analysiert und durch entsprechende Vorkehrungen die Wahrscheinlichkeit ihres Wiedereintretens reduziert. Dies ist ein Prozeß, der gesteuert werden muß, sei es innerhalb eines Applikationsgebietes, eines Unternehmens oder innerhalb einer ganzen Branche. Die Chancen sind umso besser, je mehr der Softwareentwicklungsprozeß als Werkzeug verstanden und je mehr das Gespräch zwischen Experten gefördert wird, um nicht nur Probleme aus den unterschiedlichen Sichten zu beleuchten, sondern auch um Erfahrungen auszutauschen und die Möglichkeit zur gemeinsamen Beschreibung von Aufgaben zu haben.

Um 1970 wurden die strukturierte Programmierung und schrittweise Verfeinerung vorgeschlagen, um eine Ordnung in die 'Chaosprogrammierung' zu bekommen. Sodann wurden für die Vorphase strukturierte Entwurfsmethoden bekannt und daraufhin die Verfahren auf die Analysephase ausgedehnt. Bei allen Entwicklungen entstanden für die aufgezeigten Methoden jeweils rechnergestützte Werkzeuge. Als wesentlicher Nachteil kann jeweils die fehlende Unterstützung bei der notwendigen Vorgehensweise in bestimmten Applikationsgebieten angeführt werden.

So sind zwar adäquate Ausdrucksmittel und Konsistenzbetrachtungen innerhalb der softwaretechnologischen Betrachtungsweise vorhanden, aber bestimmte Klassen von Problemen können nicht einfach ohne Kenntnisse der fertigungstechnischen Seite behandelt werden (z.B. Fragen zur Koordination und Synchronisation, zum Zeitverhalten und Durchsatz bei Montagesystemen). Als Grundlage der Verbindung beider Seiten (Ingenieur- und Informatikwissen) wird - wie im Kap. 4 erwähnt - der ADT herangezogen und als geeignetes Strukturierungs-/Modularisierungskonzept betrachtet (Bild 37).

Nach /96/ wird der ADT wie folgt definiert:

> *"Ein abstrakter Datentyp definiert eine Klasse von Datenobjekten. Die Klasse wird ausschließlich und vollständig charakterisiert durch das äußere Verhalten der Objekte. Dieses Verhalten wird mit einer Menge von Operationen (Funktionen) beschrieben, die über Exemplaren der Klasse erlaubt und erklärt sind. Die Operationen definieren die Klasse nicht nur, sondern sind auch gleichzeitig die einzige Möglichkeit auf Realisierungen des Typs zuzugreifen. Dem Benutzer des Datentyps bleibt unbekannt, wie jedes Objekt repräsentiert ist und wie die Operationen intern wirken."*

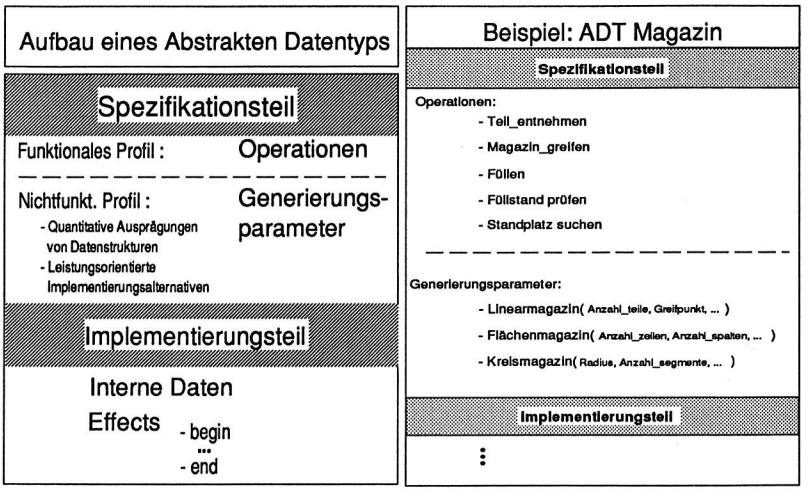


Bild 37: Abstrakter Datentyp aus Anwendersicht mit Beispiel

Der ADT stellt eine Erweiterung der Elementartypen dar und läßt genauso wie diese auf der durch ihn repräsentierten Datenstruktur nur exakt festgelegte Operationen zu. Dieses Dualitätsprinzip von Datenstruktur und zugehörigen Operationen ist das entscheidende Merkmal der objektorientierten Softwareerstellung.

Der ADT ist kein festdefinierter Objekttyp (Integer, Real) mit festgelegten Operationen (Addition, Subtraktion), sondern ein Mechanismus, der einem Benutzer erlaubt, eigene Datentypen mit entsprechenden Operationen zu definieren (z.B. 'Magazin', 'Station').
Der Nutzen von 'Abstrakten Datentypen' liegt im Strukturierungsprinzip, und dies auch in bezug auf ein bestimmtes Anwendungsgebiet. Im Gegensatz dazu bieten übliche Programmiersprachen nur einen festen, vom Anwendungsgebiet unabhängigen Satz

elementarer Datentypen an. Mit Hilfe von ADTen kann man charakteristische Begriffe eines speziellen Anwendungsgebietes eigens definieren. Ein Anwender kann mit diesen Begriffen so arbeiten, als wären diese bereits von einer Spezifikationssprache bzw. Programmiersprache zur Verfügung gestellt, und somit auch von der Implementierung abstrahieren. Er verwendet die vordefinierten ADTen als Softwarechips und kann sich so ganz auf die Lösung seiner eigentlichen Problemstellung konzentrieren.

Weiterhin sind bestimmte Fähigkeiten, z.B. Generierungsparameter, ein wesentliches Merkmal der ADTen. Mit Hilfe der generischen Parametrisierbarkeit kann ein Anwender die ADTen den Anforderungen seiner speziellen Applikation individuell anpassen, z.B für den ADT 'Magazin' die Anordnung und Füllmenge. Software, die auf der Basis von ADTen erstellt wird, beachtet also Aspekte einer möglichen Modifizierbarkeit und einer Wiederverwendbarkeit.

Eine weitere Forderung, die Trennung von Spezifikation und Implementierung, wird durch den ADT erfüllt. So lenkt eine Verwendung des Erscheinungsprofils von Objekten in einer konkreten Applikation nicht von der globalen Problemstellung ab. Darüberhinaus kann durch dieses Konzept eine Sammlung (Bibliothek) aufgebaut werden, deren Struktur und Elemente auf ein Anwendungsgebiet abgestimmt sind, und dort wiederverwendbare Einheiten (einfache Objekte, Teilsysteme) zur Verfügung stellen (Bild 38).

Im Moduldesign geht es darum, ADTen systematisch zusammenzufassen. Ein Gesamtebenenmodell unärer ADTen kann meist aus der Hierarchie der ermittelten Anforderungen abgeleitet werden. Die Erstellung dieser Ebenenhierarchie ist automatisierbar und erfolgt nach der Relation "Benutzt Objekte eines (unären) ADT zur Implementierung". Das Moduldesign stellt dagegen eine kreative Tätigkeit dar. Zwar ist auch beim Moduldesign ein methodisches Vorgehen nach bestimmten Kriterien möglich (und unerläßlich); die Entscheidung, welche Operationen zu welchem ADT gehören sollen, trifft aber letztlich der Designer (evt. rechnerunterstützt). Entscheidend beim Moduldesign ist die Aufgabe, sowohl Kriterien der Softwaretechnologie als auch Kriterien des Applikationsgebiets, d.h. in diesem Fall der "Flexiblen Montage", bei der Durchführung zu berücksichtigen.

Die softwaretechnologischen Kriterien für das Moduldesign sind Lokalität, Abstraktion, Objektorientierung und Hierarchisierung /14/. Durch die Anwendung dieser Strukturierungsprinzipien werden softwaretechnologische Ziele wie Zuverlässigkeit, Verständlichkeit, Transparenz, Adaptabilität, Portabilität gefördert sowie die Verifizierbarkeit wesentlich erleichtert. Nach dem Lokalitätsprinzip sollen sich alle Stellen, von denen aus ein

Objekt manipuliert wird, und alle Kontrollinformationen, die die Manipulation regeln, in örtlicher Nähe des Objektes selbst befinden /54/. Dies besagt, daß Manipulationen eines Objektes eines ADT nur über die Operationen möglich sein sollen, die innerhalb des ADT definiert sind und die Synchronisation der Zugriffe auf ein Objekt im Objekt selbst stattfinden soll.

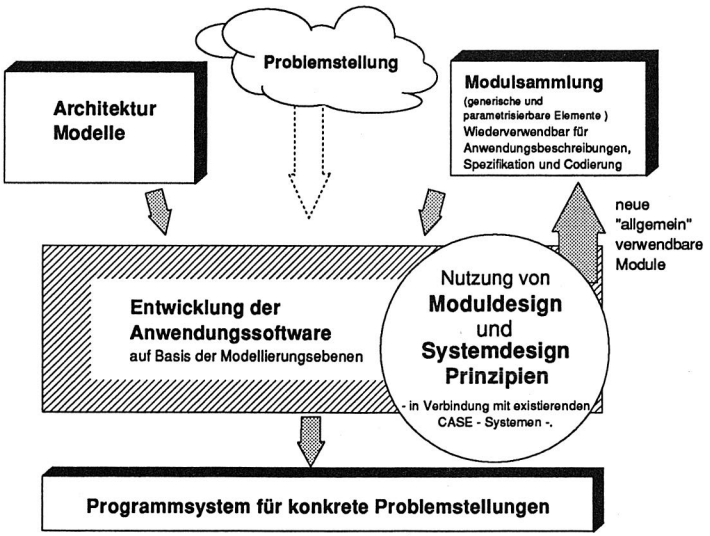


Bild 38: Vorgehensweise bei Applikationssoftware-Erstellung

Aspekte und Spezifika des Applikationsgebiets müssen in die Überlegungen des Moduldesign miteinbezogen werden, damit die abgeleiteten ADTen den Erfordernissen des Applikationsgebiets wirklich entsprechen. Hierarchische Steuerungsstrukturen zeigen, daß auch in diesen Bereichen die (softwaretechnologischen) Kriterien Hierarchisierung, Modularisierung, Abstraktion als relevante Strukturierungskonzepte erkannt wurden, um die Komplexität der vorhandenen Problemstellungen zu bewältigen. Diesen Strukturierungsbestrebungen soll natürlich auch beim Moduldesign Rechnung getragen werden.

Die Klassifizierung der Anforderungen (Funktionen) einer Zellensteuerung durch Abstraktionsebenen, wie sie das hierarchische Steuerungskonzept vorsieht, soll auch bei der Gestaltung der ADTen Eingang finden (Bild 38). Anforderungen, die zu verschiedenen Ebenen dieser Hierarchie gehören, sollen daher auch zu verschiedenen ADTen gehören.

6. Informationsstrukturen in den Ebenen

Im Mittelpunkt steht weiterhin der Grundgedanke, durch schrittweise Abstraktion vermehrte Geräteunabhängigkeit auf den verschiedenen Stufen der Betriebsmittelhierarchie zu schaffen. Ziel dieses Ansatzes ist es, über die Parametrisierung der logischen Arbeitsplanelemente in der zugeordneten Abstraktionsstufe und über die Einbeziehung von Zustandsdaten die Generierung der benötigten Steuerungsvorgaben zu ermöglichen und somit die Flexibilität der Montagezelle zu erhöhen. Eine systematische Zuordnung relevanter Datenbereiche dient als Ausgangspunkt zur Definition von Schnittstellen innerhalb des Ebenenmodells, wodurch letztendlich auch die Bemühung um eine Standardisierung (Begriffsabgrenzung) im Bereich der Montagesteuerungen unterstützt und die notwendige Transparenz (z.B. durch Schaffen eines konzeptionellen Datenschemas) erreicht wird.

6.1 Relevante Datenbereiche

Bei der Gestaltung flexibler Informationsstrukturen in einer FMZ wird zunächst vom Begriff der Produktfamilie ausgegangen /66, 98/, die einer Menge von Erzeugnissen mit derselben globalen Montagestruktur entspricht. Varianten ergeben sich über den Austausch einzelner Bauteile/Baugruppen und über die Variation der Anzahl von Teilen/Baugruppen.

Die Variantenanzahl in einer Produktfamilie kann sehr groß werden, wie das Modellprodukt 'Telephon' aus Kap. 7 zeigt: die Auswahl aus 4 Gehäusen, 6 Platinen, einem 4x3 Rastergrundelement und 3 Tastenkörpern mit 15 möglichen Tastenausprägungen ergibt dann 4 x 6 x 1 x 3 x 15 ** 12 Varianten; (vgl. Produktvielfalt bei Telefonapparaten).

Als Ausgangspunkt für die Strukturierung relevanter Informationen werden jeweils folgende Datenbereiche betrachtet, wobei konkrete Datenelemente mehreren dieser Bereiche zugeordnet sein können: Stamm- und Strukturdaten, Produktstruktur- und Arbeitsplandaten, Zustandsdaten, Auftrags- und Materialdaten.

6.1.1 Stammdaten und Arbeitsplandaten

Die Stamm- und Strukturdaten tragen zur statischen Beschreibung der Systemkonfiguration bei und sind unabhängig von Aufträgen und konkreten Arbeitsplänen. Die Strukturdaten stellen hierbei eine Untermenge dar und kennzeichnen vorallem Daten, die aufgrund neuer Vorgaben entstehen (z.B. Werkstückdaten, Arbeitsplan) /40/, aber dann

innerhalb der FMZ eine feste Größe bilden. Zum Aufbau des Rahmenarbeitsplans werden ausschließlich Stammdaten und parametrisierte Strukturdaten herangezogen.

Sie bilden die Basis für die Abwicklung von Montage-, Transport- und Hilfsoperationen in einer Zelle. Als Träger technologischer Informationen unterstützen sie die Steuerung bei der Bestimmung des "Wie?" für die Durchführung eines Montagevorgangs. Ihre Verwaltung erfolgt auf den unterschiedlichen Abstraktionsebenen. Unter anderem zählen dazu:

- Teilestammdaten: Geometrie, Fügestellen, Gewicht, Material
- Stationsdaten: Beschreibung der technologischen Fähigkeiten bzw. Grenzen
- Werkzeugdaten: Bezeichnung und technologische Parameter
- Strukturdaten: Beschreibung des Zellenaufbaus, Verknüpfung der Zellenorte etc.
- Werkstückträgerdaten: Palettenbeschreibung und Zuordnung der transportierbaren Teile
- Bereitstellungsdaten: Magazinbeschreibungen und Zuordnung von Teilen
- Bereitstellungsplatzdaten: Zuordnung von Orten zur Materialhaltung im Arbeitsbereich
- Fähigkeitsdaten: Rahmenarbeitspläne, logische Arbeitsgänge und Arbeitsschritte

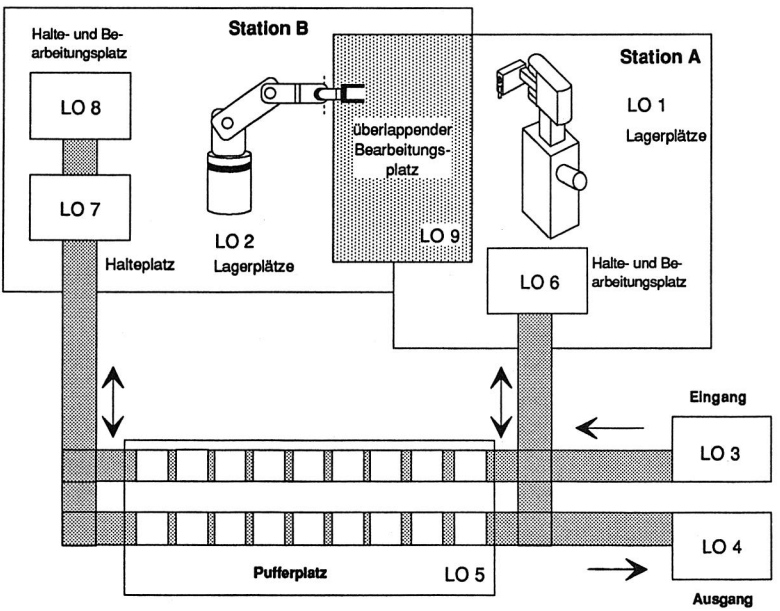


Bild 39: Beispielhafte Zellenaufteilung in 'Logische Orte' (LO): Basisdaten der Ebene 5

Das Prinzip der **'Logischen Orte'** (Bild 39) unterscheidet neben den Stationen für die Ebene 6 eine Charakterisierung globaler logischer Orte (LO) für die Ebene 5 und der lokalen logischen Orte (LLO) für die Ebene 4. Funktional zusammengehörige Orte werden zu globalen logischen Orten zusammengefaßt, charakteristisch hierfür sind:

- eindeutige Identifizierung (z.B. durch Nummern)
- Zuordnung (z.B. Transportsystem, Station)
- Typ (Puffer, Bearbeitungs-/Lagerungsplatz, Ein-/Ausgang)
- Kapazität (Menge an Einheiten, z.B. Paletten)
- Verbindungsaufbau (Kopplung der einzelnen LOs)

Zur Darstellung des Verbindungsaufbaus gibt es unterschiedliche Möglichkeiten, z.B einen Verweis auf die Verbindungsmatrix (Bild 40) oder die explizite Angabe der Kopplungen. Die logischen Orte einer Zelle sind untereinander entweder direkt, über andere logische Orte oder überhaupt nicht verbunden. Mit einer Matrix Vn zeigt man alle Verbindungen ('1') auf, die mit <= n Schritten erreicht werden. So ergibt sich bei jeder Konstellation eine Matrix V = Vn /68/, die alle möglichen Verbindungen der LOs einer Zelle angibt.

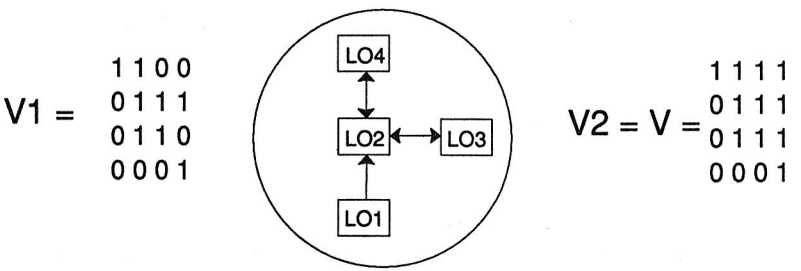


Bild 40: Matrixdarstellung der Verbindungen 'Logischer Orte'

Bei expliziter Wegaufzählung - mit Aufführung evt. Zyklen - gilt dann für das Beispiel:

Wege : { (1,2), (1,2,3), (1,3), (1,4), (2,3), (2,4), (3,2), (3,2,4) }

Zyklen : { (2,3,2) }

Durch die Einführung lokaler logischer Orte (LLO) können die Arbeitsplandaten stationsunabhängig gehalten werden, d.h. es wird eine stationsrelative Identifikation eingeführt. Bei dieser Lösung erfolgt also die Abbildung der funktionalen Orte einer 'Norm-

station' auf die vorhandenen logischen Orte der einzelnen Montagestationen. Es ist zu beachten, daß jeder LO gleichzeitig mehreren funktionalen LLOs zugeordnet sein kann. Umgekehrt ist jeder LLO für jede Station einem anderen LO zugeordnet. Zwischen den Begriffen existiert also eine n : m Beziehung. Charakteristisch für lokale logische Orte ist also:

- mehrdeutige Identifizierung (Stationsunabhängigkeit)
- eindeutige LO-Zuordnung für jede Station
- Einschränkungen bzgl. des Beladetyps
  (z.B nur bestimmte Teileklassen aufnehmbar)

Aufbauend auf den Fähigkeiten der einzelnen Montagestationen und der Gesamtzelle liefert der auftragsneutrale Arbeitsplan, genannt Rahmenarbeitsplan, die Grundlage für die Beschreibung des Arbeitsablaufs bei den Varianten einer Produktfamilie. Somit ist jeder Produktfamilie genau ein gültiger RAPL zugeordnet, dessen Daten unabhängig von den tatsächlichen Ausprägungen der Aufträge sind. Das bedeutet, daß der RAPL weder stationsabhängig noch auftragsbezogen ist. Als Darstellungsform für Arbeitsgangfolgen wird der Strukturgraph herangezogen. Der Strukturgraph eines auftragsneutralen bzw. auftragsabhängigen Arbeitsplanes ist eine Netzstruktur, dessen Knoten Arbeitsgänge einer Produktfamilie bzw. einer Produktausprägung sind und dessen Kanten die Übergänge von einem Arbeitsgang zu einem anderen darstellen (Erläuterungen in 6.2).

6.1.2 Zustands-, Auftrags-, Materialdaten

Die Summe aller Zustandsdaten spiegelt jeweils den dynamischen Charakter der gesamten Zelle, einschließlich der Steuerungsattribute wider; die Informationsgewinnung erfolgt über MDE/BDE und Auswerteroutinen (vgl. 4.4). Zum einen dienen diese Daten der Überwachung des Montage- und Steuerungsablaufs; andererseits ergibt die Parametrisierung der Steuerungsfunktionen nur dann einen Sinn, wenn aktuelle Werte (z.B. Füllstände von Magazinen) bei der Parametergenerierung und zu Optimierungzwecken herangezogen werden. Neben Angaben zum Zustand der Geräte/Stationen werden auch Statusinformationen zu den Elementen der aktuellen Aufträge und Arbeitspläne (zu AGs und ASs etc.) geführt. Der Materialbestand auf den Abstraktionsstufen sowie die Zuordnung von Teilen zu Aufträgen gehört ebenso zur Zustandsbeschreibung.

Die Form einer Auftragsbeschreibung muß dem Ziel angepaßt werden, die Auftragsdaten als Grundlage für die Parametrisierung des zugehörigen RAPL zu nutzen. Aus der globalen Beschreibungsstruktur wird mit Hilfe der Auftragsspezifikation eine Erzeugnis-

struktur für genau diesen Auftrag generiert (z.B. mit Entscheidungstabellen). Für die Zellensteuerung dient die erstellte Erzeugnisstruktur als Ausgangspunkt zur Bestimmung der notwendigen Montagevorgänge und zur Ermittlung des Teilebedarfs. Zu einem bestimmten Zeitpunkt können sich - theoretisch betrachtet - beliebig viele Aufträge zu einer oder mehreren Produktfamilien in Bearbeitung befinden. Im wesentlichen lassen sich die auftragsbeschreibenden Daten in drei Teile gliedern:

- allgemeine Auftragsdaten
- produktfamilienspezifische Angaben
- Erzeugnisstrukturinformationen

Unter die allgemeinen Auftragsdaten fallen alle Angaben, die unabhängig sind von den Produktfamilien, z.B. typischerweise Menge, Zeitvorgaben, Prioritäten oder die Zuordnung zu einer Produktfamilie. Auftragsdaten, die nur in Abhängigkeit von der zugeordneten Produktfamilie sinnvoll sind und nicht implizit aus der Erzeugnisstruktur hervorgehen, können in den produktspezifischen Angaben aufgelistet werden. Für jeden Auftrag muß eine exakte, der Montage angepaßte Beschreibung der Produktstruktur existieren, die als Erzeugnisstrukturinformation abgelegt wird (Lösungsansatz siehe Kap. 6.2.1). Sie dient als Grundlage für:

- die Ermittlung relevanter auftragsbezogener Arbeitsgänge
- die Ermittlung des Materialbedarfs
- die Zuordnung von physisch vorhandenen Teilen in der Zelle zu den logischen Elementen des Produkts
- die Zuordnung von logischen Produktelementen zu den Inputteilen für die auszuführenden Arbeitsgänge
- die Generierung von Parametern in der Arbeitsschrittliste

Eine Klassifizierung und Bezeichnung von Montageteilen soll eine Gruppierung ermöglichen. Kriterien (Grenzwerte, etc.), für eine bestimmte **Teilklasse** (Bild 41) können sein:

- die geometrische Struktur des Teiles,
- die Magazinierung des Teiles,
- die Transportierbarkeit des Teiles,
  (Einfluß von Größe und Form auf Pallettenausprägung).

Aber erst der sogenannte **Teiltyp** gibt Einzelheiten zur Ausprägung des Teils an. Durch den Strukturtyp z.B. wird die Farbe angegeben; durch den Palettentyp werden dem Teil Transportmöglichkeiten für diese bestimmte Montagezelle zugewiesen. Durch einen angegebenen Magazintyp, d.h. durch den Ort der Resourcen, kann man einem Teil spezielleInformationen (Entnahmeart, Verwaltungsspezifika) zuweisen.

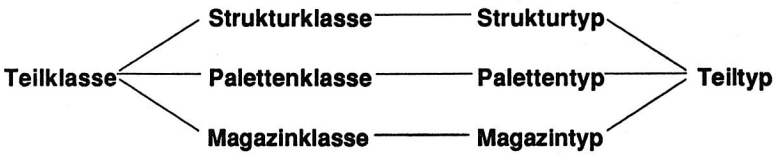


Bild 41: Aufteilung Teilklasse und Teiltyp

Es handelt sich hierbei um ein zweistufiges Verfahren. Die Bezeichnung eines Montageteils setzt sich zusammen aus einer Teilklasse und einem Teiltyp als Verfeinerung. Die Verwendung von Teilklassen unterstützt die Verwaltung bezüglich der Struktur eines bestimmten Produktes. Außerdem können gemeinsame Eigenschaften zu Transportierbarkeit und Magazinierung dem Begriff Teilklasse zugeordnet werden (Vermeidung von Redundanzen). Die Klassifizierung ist über alle Ebenen gültig; die damit verbundenen Teilestammdaten (z.B. Geometriedaten) sind aber nur auf ganz bestimmten Hierarchiestufen von Bedeutung. Grundsätzlich werden zwei Arten von Teilklassen nach dem Verwendungszweck unterschieden: **Werkstücke** und **Verbrauchsmaterial**. Erstere sind jeweils einzeln identifizierbar, während das Verbrauchsmaterial jeweils nur gesamtmengenmäßig charakterisiert werden kann.

6.2 Methoden zur Darstellung der Informationen

6.2.1 Knotentypenmethode zur Darstellung der Produktstruktur

Als Lösungsansatz wird eine an /71, 112/ angelehnte Darstellungsweise gewählt. Den Ausgangspunkt der Knotentypenmethode bildet eine globale Produktstruktur, die den montageorientierten Aufbau der Varianten innerhalb einer Produktfamilie beschreibt. Weiterhin wird ein Algorithmus vorgestellt, der die Reduzierung der globalen zur auftragsabhängigen Produktstruktur aufzeigt.

Knotentypen (Grundelemente)

Teileknoten repräsentieren ein oder mehrere Ausgangsteile derselben Klasse für die Montage. Die Anzahl n, mit der das Teil in eine Baugruppe eingeht, ist durch die variable Mengenangabe an der zugeordneten Kante gegeben, und die Information zur Teilklasse und späteren Ausprägung (Teiltyp) hängt ebenfalls am Knoten. Für die globale Erzeugnisstruktur muß weder n noch der Teiltyp festgelegt sein.

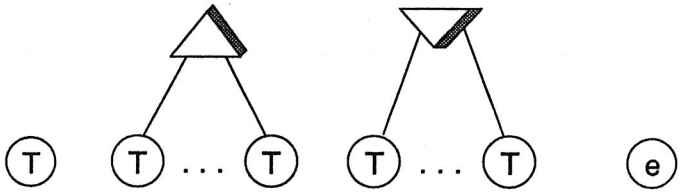


Bild 42: Knotentypen

Konjunktivknoten verbinden alle Knoten, die Teile repräsentieren, die auf dasselbe Basisteil (ein ausgezeichneter Unterknoten) montiert werden (vgl. Bild 42). **Alternativknoten** ermöglichen über die Wahl eines Zweiges Produktvarianten oder die Verwendung unterschiedlicher Teile bei der Montage einer Variante (z.B. fertige 'Zwischenbaugruppe' statt Einzelteile). **Leerknoten** existieren, um sogenannte Kann - Varianten bei der Alternative zu ermöglichen (z.B. Verbindungsknopf mit Funktionsplatine beim Telefon).

Abhängig von einem Auftrag wird die globale Produktstruktur (nur mit Teilklassen) in eine auftragsbezogene Struktur umgewandelt. Die Teileanzahl wird hierbei den Knoten variabel (auftragsabhängig) zugeordnet. Diese sogenannte Reduzierung der globalen Produktstruktur kann in zwei Schritten geschehen:

Schritt 1:

Auf Basis der Auftragsbeschreibung werden zunächst alle Alternativknoten aufgelöst. Die Teileanzahl n jedes Teileknoten wird genau bestimmt und gemäß der gewünschten Ausgangsteile wird dieser Teileknoten durch n Teileknoten ersetzt. Jedem Teileknoten wird nun der zugehörige Teiltyp zugeordnet, wobei die Zuordnung der Teiltypen nach festen Regeln (z.B. von links nach rechts) erfolgt. Das Ergebnis beschreibt genau eine Ausprägung eines konkreten Auftrages.

Schritt 2:

Für die Weiterverarbeitung der Erzeugnisstruktur werden alle Basisteilknoten mit ihren Konjunktivknoten identifiziert. Die Leerknoten mit ihren Kanten fallen weg.

Die entstandene Datenstruktur kann der Zelle zusammen mit dem Auftrag übergeben oder in autarken Systemen aber auch erst entwickelt werden. Sie bildet dann eine Grundlage für weitere Steuerungsmechanismen.

Beispiel

In Bild 43 besteht die Produktfamilie 'Telefon' (vgl. Kap. 7.1) aus einem Gehäuse (A), 1 oder 2 Platinen (B), einer Tastatur aus Grundkörper (C) mit einer nx3-Anordnung und mindestens 3 Tasten.

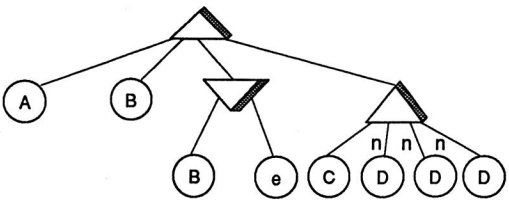


Bild 43: Produktfamilie

Es soll ein grünes Telefon mit einer Platine vom Typ 1 und einer 1x3-Tastatur mit rot, weiß, roter Tastenfeldanordnung ausgestattet werden. Die Reduzierungsschritte sind in Bild 44 dargestellt.

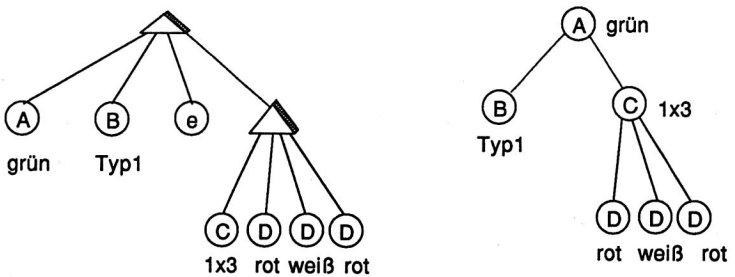


Bild 44: Reduzierungsschritte 1 und 2

6.2.2 Aufbau und Struktur von Rahmenarbeitsplänen

Aufbauend auf den Anforderungen aus Kapitel 3 werden Grundlagen für Strukturen und Elemente eines Rahmenarbeitsplanes (RAPL) angesprochen, die zur Beschreibung des Arbeitsablaufs der Varianten einer Produktfamilie nötig sind. Hierbei spielt aus der Sicht der organisatorischen Freiheiten und der Generierungsmechanismen in der Zelle der Begriff '**Arbeitsgang**' die entscheidende Rolle . Es wird nun erläutert, wie ein dreistufiger Aufbau des Arbeitsgangbegriffes einen Beitrag zu Flexibilitätsansätzen leisten kann.

1) Logische Arbeitsgänge (LAG):

LAGs sind unabhängig von den Rahmenarbeitsplänen in der Zelle. Sie beschreiben allgemein Fähigkeiten und zählen zum Bereich der Stammdaten. Jeder LAG entspricht einer bestimmten Abfolge logischer Arbeitsschritte. Mit dem Begriff des LAG ist auch die Ausführbarkeit an bestimmten Stationen mit bestimmten Werkzeugen verbunden, d.h. er bildet die Basis für die Arbeitsvorbereitung und dient zur Bildung einer Einheit von technologisch orientierten Arbeitsschritten.

> Man beachte: Indem logischen Arbeitsschritten bestimmte Technologien zugeordnet werden (z.B. einem Fügeschritt das Verfahren 'Fügen durch Einlegen'), definiert der LAG die Grobstruktur eines Montageabschnittes.

2) Arbeitsgänge (AG):

AGs sind Ausprägungen von LAGs innerhalb des RAPL einer Produktfamilie, d.h. für jeden RAPL können mehrere AGs als verschiedene Ausprägungen eines einzigen LAGs existieren. Sie repräsentieren dann notwendige Montageabschnitte. Zwischen den AGs im RAPL und den Knoten der globalen Erzeugnisstruktur der Produktfamilie besteht eine feste Zuordnung. Somit existieren neben einem Bezug zu einem LAG Inputdaten (Basisteil, Fügeteile, Verbrauchsmaterial) mit Verweisen auf Knoten der Erzeugnisstruktur und Outputdaten mit Verweis auf einen Knoten der Erzeugnisstruktur.

3) Arbeitsgänge im Strukturgraph des RAPL (RAP-AG):

Zur Koordinierung der Arbeitsgänge existiert als Teil des Rahmenarbeitsplans ein Graph (Vorranggraph) zur Beschreibung erlaubter Folgen von AGs. Jeder Knoten dieser Struktur entspricht einem RAP-AG. Verschiedene RAP-AGs können dabei denselben AG repräsentieren. Durch den Begriff RAP-AG ist es möglich, sowohl den Output nach Fertigungstufe zu unterscheiden als auch das eigentliche Outputteil eines AGs nach seiner Klasse und Typ eindeutig zu bestimmen.

Der Strukturgraph (Vorranggraph) stellt sämtliche Variationsmöglichkeiten der Arbeitsgänge im Montageablauf für die ganze Produktfamilie als Netzdatenstruktur dar. Für die gegebene Problemstellung wird nun der gerichtete Graph benutzt. Dieser Graph ist ein Tripel G = (P,K,v), bestehend aus einer Menge P von Knoten, einer Menge K von Kanten und einer Funktion v, die jeder Kante aus K ein geordnetes Paar von Knoten aus P zuordnet (Definition in /77/).

Die Knoten kennzeichnen einen RAP-AG und die Kanten jeweils den Übergang von einem RAP-AG zum nächsten. Eine Folge von Kanten und Knoten nennt man Pfad mit

jeweiligen Vorgänger- und Nachfolgerknoten. Die Knoten (RAP-AG) werden durchnumeriert und im jeweiligen Knoten steht der dem RAP-AG zugeordnete AG. Beim Aufbau des Strukturgraphen besitzen die Knoten (RAP-AG) verschiedene Attribute: einfach, optional, leer (Bild 45). Der optionale Knoten kann aufgrund der Erzeugnisstruktur wegfallen, d.h. er wird irrelevant. Der leere Knoten wird als Hilfsknoten bei bestimmten Anordnungen (z.B. ersetzende RAP-AG-Pfade) eingesetzt.

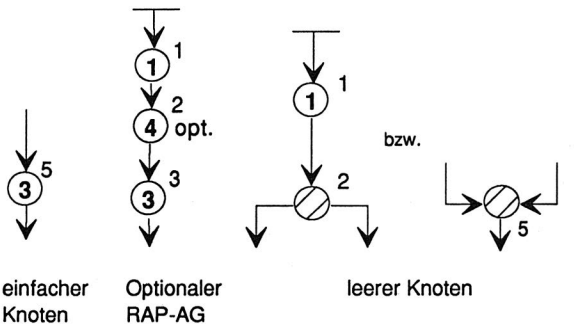


Bild 45: Knotenattribute

Die Anordnungselemente des Strukturgraphen

Bei der Abfolge der Knoten im Graphen unterscheidet man die Anordnungselemente: sequentiell, ersetzend, alternativ, parallel (vgl. Bild 46).

Bei **sequentiellen** RAP-AG-Folgen existiert für den Vorgänger- und Nachfolgerknoten jedes RAP-AG eine eindeutige Abbildung. Mit **ersetzenden** RAP-AG-Folgen stellt man die Auswahl zwischen verschiedenen RAP-AG-Pfaden dar. Auftragsbezogen wird später der entsprechende RAP-AG-Pfad entnommen. Die übrigen Pfade werden dann irrelevant und aus dem RAPL gestrichen.

Alternative RAP-AG-Folgen bezeichnen Abarbeitungsreihenfolgen, die variieren können. Diese Folge kann selbst als Menge betrachtet werden, auf der z.B. bei Prioritätenvorgabe auch eine Ordnung liegen kann. Im Gegensatz zur ersetzenden Folge hängt die Entscheidung der Auswahl nicht von der Erzeugnisstruktur ab, sondern von aktuellen Zuständen, Optimierungskriterien und Entscheidungen des Bedienpersonals.

Parallele RAP-AG-Folgen bezeichnen in einem erweiterten Strukturgraphen parallele Möglichkeiten und Zwänge. Sie können dann parallel ausgeführt werden, wenn kein

Arbeitsgang irgendwie abhängig ist von Ereignissen und Ergebnissen der anderen Arbeitsgänge (z.B. Zusammenbau unterschiedlicher Baugruppen). Parallele RAP-AG-Folgen können jederzeit auch sequentiell abgearbeitet werden.

Der Parallel-Zwang hingegen kann bei Einbeziehung der Synchronisation von Arbeitsgängen auftreten (z.B. Problematik bei Doppelrobotereinsatz). Im Strukturgraphen werden diese AGs, die auf Ebene 5 und 4 synchronisiert werden müssen, als ein Knoten dargestellt.

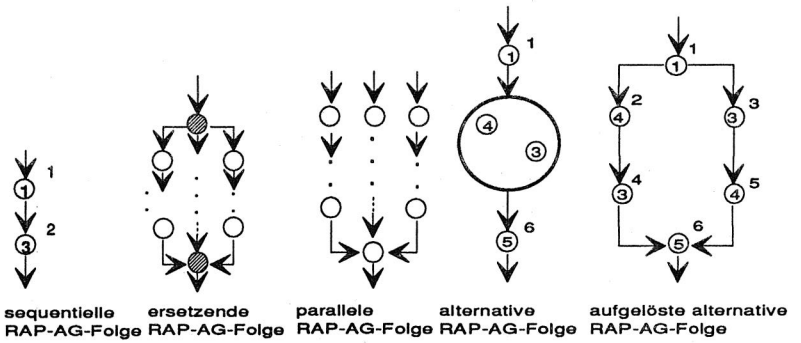


Bild 46: Knotenfolgen

Innerhalb des Strukturgraphen müssen nun folgende Informationen zu den jeweiligen Arbeitsgängen gespeichert sein:

- der Bezug auf einen LAG
- die Anzahl der Wiederholungen für die einzelnen Arbeitsschritte und für die Schrittfolge (Konstante oder Verweis auf die Teileanzahl eines Erzeugnisstrukturknoten).
- Inputdaten mit Verweisen auf einen oder mehrere Knoten der Erzeugnisstruktur. Basisteil, Fügeteile und Verbrauchsmaterial werden dabei unterschieden.
- Outputdaten mit Verweis auf einen Knoten der Erzeugnisstruktur (Teilebezeichnung vom RAP-AG abhängig).

Für jeden RAP-AG sind folgende Angaben zu speichern:

- zugeordneter AG
- Fertigungsstufe
- optional?
- Unterscheidung des Outputtyps (Fertigteil, Baugruppe, Zwischenprodukt)
- Daten zur Benennung des Outputteils nach Klasse / Typ

6.2.3 Freiheitsgrad: Station und Ablauf

Bei stationsunabhängiger Beschreibung des Arbeitsplanes sind Informationen über Teileidentifikation sowie über Ausgangs- und Fertigteile und Angaben über die Koordinierung von AGs notwendig. Die Benennung der Input-/ Outputteile wird für jeden RAP-AG festgelegt unter Anwendung von Methoden der angewandten Mathematik und der Einhaltung gewisser Ordnungsprinzipien /111/, die einen Bezug auf einen FA, auf ein konkretes Zwischenprodukt bzw. konkrete Baugruppe herstellen oder sogar eine eineindeutige Bezeichnung (z.B. Numerierung in Verbindung mit einem FA in der FMZ) garantieren. So gilt: Für i=1..N: $I_{ij} \wedge O_i = \{\}$.

Seien hierbei RAP-AG_i (i=1,..,N) die Arbeitsgänge des betrachteten Arbeitsplanes. Zu jedem AG sei I die Menge der zur Bearbeitung benötigten Teile und O das entstehende Teil. Die Menge I_i der Inputdaten besteht aus Teilen, die aus der Rahmenstückliste und aus den Outputteilen der vorhergehenden RAP-AG's (z.B. nach parallelen Folgen) stammen. In der Rahmenstückliste muß für alle aufgelisteten Teile eine Zuordnungsreihenfolge zu der Menge von Inputdaten existieren. In der Zellensteuerung kann dann mit mathematischen Methoden über Ausgangs- und Fertigteile (z.B. für Bestellungen und Abtransporte) sowie über Zwischenprodukte Information gewonnen werden. Die Koordinierung zwischen Arbeitsgängen - mit Ausnahme der "Koordinierung wegen Blockierung" - kann dann auf die Koordinierung zwischen einem AG und dem Vorhandensein von Teilen reduziert werden.

Die Forderung nach Stationsunabhängigkeit einerseits und die stationsspezifischen Angaben andererseits können durch zwei unterschiedliche Lösungswege 'Stationsunabhängigkeit durch Fähigkeitsinformationenen der Stationen' und 'Stationsunabhängigkeit durch ersetzende Stationen' erfüllt werden. Beide Wege und deren Auswirkungen werden nun aufgezeigt.

Stationsunabhängigkeit durch "Fähigkeiten" der Stationen

Der erste Weg sieht vor, im Arbeitsplan keinerlei Informationen über Stationszuordnung eines AGs aufzunehmen. Die stationsspezifischen Daten für AGs stehen in einer Wissensbasis als Stammdaten zur Verfügung. Der Begriff des AGs wird demnach stationsunabhängig verstanden (vergl. die Dreiteilung des Arbeitsgangbegriffes).

Im allgemeinen werden in einer FMZ nicht alle Stationen die gleichen Tätigkeiten verrichten können. Es ist nun also für jeden Montagevorgang eine Liste der Stationen aufzu-

stellen, die die nötigen "Fähigkeiten" haben, den AG auszuführen. In dieser Liste sind dann auch die stationsspezifischen Daten (RC-Programm(e), Werkzeuge/Greifer) zu führen.

Beispiel:

Gegeben sei z.B. eine FMZ mit vier Stationen, zwei AGs ("AG1" und "AG2") und die dazugehörigen Fähigkeitslisten:

AG1:

Station 1: Fähig = "ja", RC-Pgm = "0101", WZ = "1111", "1112"
Station 2: Fähig = "nein"
Station 3: Fähig = "ja", RC-Pgm = "0103", WZ = "1113", "1135"
Station 4: Fähig = "nein"

AG2:

Station 1: Fähig = "nein"
Station 2: Fähig = "nein"
Station 3: Fähig = "nein"
Station 4: Fähig = "ja", RC-Pgm = "0104", WZ = "2244"

Stationsunabhängigkeit durch ersetzende Stationen

Der zweite Lösungsweg sieht vor, im Arbeitsplan die möglichen Stationen zur Ausführung eines AGs anzugeben - mit evt. Auswahlkriterien - und gleichzeitig auch stationsspezifische Daten mitzuführen. Der Arbeitsplan ist so zwar an eine Angabe konkreter Stationen gebunden, es verbleibt aber genügend Raum für flexible Zuordnung einer ausführenden Station zu einem Arbeitsgang. In der Realität ist die Anzahl der für einen Arbeitsgang in Frage kommenden Stationen ohnehin begrenzt.

Bewertung beider Verfahren

Das erste Verfahren erlaubt eine maximal flexible Zuordnung der ausführenden Station. Neu hinzukommende Stationen oder neue Fähigkeiten vorhandener Stationen können ohne Änderung eines Arbeitsplans genutzt werden. Auf Maschinenausfälle kann flexibel reagiert werden, indem die entsprechenden Fähigkeiten 'gelöscht' werden.

Das zweite Verfahren ist wesentlich einfacher und den Gedankengängen eines Planers angepaßt. Es erlaubt außerdem dem Ersteller des Arbeitsplans, eine Station bevorzugt für einen Arbeitsgang einzusetzen, bevor notfalls auf eine ersetzende Station ausgewichen wird. Alle anzugebenden stationsspezifischen Daten in bezug auf den Arbeitsplan sind hierbei zusammengefaßt.

Der Nachteil des ersten Verfahrens ist, daß bei der Einführung neuer AGs auch entsprechende Fähigkeitslisten neu erstellt werden müssen. Der Aufwand hierfür kann enorm wachsen, wenn ein AG z.B. Kombinationen von existierenden Folgen von RC-Programmen entspricht. Eine der wesentlichen Ideen des zugrundeliegenden Gesamtkonzepts ist es aber, für die IR-Steuerung nur Rumpf-RC-Programme (-> Arbeitsschritte) bereitzuhalten und durch deren geschickte Parametrisierung und logische Verknüpfungen eine Vielfalt von Montagevorgängen abzudecken /55, 60/. Eine Lösung durch Fähigkeitslisten schränkt also den hier verfolgten Weg ein. Im weiteren wird daher die Methode der "ersetzenden" Stationen verfolgt und der Begriff der Normstation aus 6.1 übernommen.

Sowohl für benötigte Teile als auch für benötigte freie Lagerkapazität muß der Ort ihrer Bereitstellung (im Arbeitsplan) spezifiziert werden. Dieser Ort ist aber offensichtlich davon abhängig, auf welcher Station der Arbeitsgang ausgeführt wird. Als Lösung werden hier die Orte zu diesem Zweck stationsrelativ bezeichnet. Funktional gleichwertige Orte haben dann für jede mögliche Station die gleiche stationsrelative Identifikation (LLO). Da die Ebene 4 die Plätze einer Station selbsttätig verwaltet, kommen als Orte für die Bereitstellung von Teilen nur in Frage

- die "Logischen Orte" im Arbeitsbereich einer Station (von der Ablaufsteuerung kontrollierten Übergabeplätze) und
- ein Handhabungs-/Montage- oder Lagerungsplatz (LO) in der Peripherie der Station (konkrete Auswahl durch Ebene 4)

Damit sich für alle AGs eine eineindeutige lokale Numerierung findet, müssen für einen "LO" evt. mehrere lokale LO-Nummern geführt werden. Es entsteht so eine **virtuelle Normstation**, die sich auf jede Station abbilden läßt (vgl. 6.1).

6.2.4 Die Reduzierung des Strukturgraphen

Mit einer Reduzierung des zugeordneten RAPL ist zunächst insbesondere die Elimination aller RAP-AG im Strukturgraph verbunden, die für den betrachteten Auftrag irrelevant sind, d.h. die für dieses Produkt nicht ausgeführt werden müssen. Aufgrund der Auftragseingabe bzw. Übergabe vom Leitrechner/PPS ist der zugehörige Rahmenarbeitsplan auszuwählen. Mit Hilfe der Stückliste und des Rahmenarbeitsplanes wird der auftragsbezogene Variantenarbeitsplan generiert. Hierbei werden die der Variante entsprechenden Arbeitsgänge aus dem Rahmenarbeitsplan ausgewählt und die entsprechenden Teiltypen eingetragen, d.h. die Systemsteuerung eliminiert erst alle für genau einen Auftrag nicht relevanten optionalen und ersetzenden RAP-AGs, indem die Daten der Stückliste mit den Inputdatenklassen der zugeordneten AG verglichen werden.

Nach der Reduzierung fallen die irrelevanten Knoten und deren dazugehörigen Kanten weg. Sowohl bei den verschiedenen Anordnungselementen als auch bei den Knotenarten des Strukturgraphen unterscheidet man zwischen solchen, die nur im RAPL existieren, wie optionale Knoten und ersetzende RAP-AG-Folgen und unter denen, die auch im reduzierten Strukturgraphen erhalten bleiben, wie einfache und leere Knoten, sequentielle, parallele und alternative RAP-AG-Folgen.

Die zweite Generierung, die konkrete Festlegung einer Arbeitsgangfolge, kann vor Ausführungsfreigabe des Arbeitsplanes für den Gesamtauftrag, für eine bestimmte Losgröβe oder je Teilauftrag auf Ebene 6 durchgeführt werden. Es entsteht jeweils ein variantenspezifischer Arbeitsplan, dessen sequentielle Arbeitsfolge aufgrund des aktuellen Zustandes der Gesamtzelle (Ebene 6) optimiert ist.

Genauso kann aber unter Beibehaltung der Freiheitsgrade der Strukturgraph auf Ebene 5 weitergeleitet werden. Hierdurch kann dann prozeβnäher und noch kurzfristiger die jeweiligen auszuführenden Arbeitsgänge unter Einbeziehung gewünschter Kriterien ausgewählt werden. In allen Fällen wird bei der Abarbeitung des reduzierten Strukturgraphen von den Alternativen jeweils nur ein Pfad ausgeführt, so daβ alle Knoten in den übrigen alternativen Pfaden irrelevant werden.

Diese Reduzierung kann auch mit Hilfe eines Entscheidungstabellenverbundes realisiert werden, denn ein entscheidender Aspekt bei der Arbeitsgangfolgegenerierung liegt darin, daβ aktuelle zellenspezifische Restriktionen zu berücksichtigen sind. Dies bedeutet, daβ betriebliche und montagetechnische Rahmenbedingungen und auch das Know-how und die Erfahrung des Systemspezialisten miteinbezogen werden können. Die Beschränkungen können übersichtlich in Entscheidungstabellen als Regeln formuliert werden. Durch jede Entscheidungstabelle wird nachgeprüft, ob Station, Material und Greifer für den auszuführenden Arbeitsgang verfügbar sind.

Die dadurch festgelegten Restriktionen entsprechen der Zielfunktion, die für die Ausführung aller AGs Gültigkeit hat. Diesen Bedingungen entsprechend wird der AG ausgewählt oder die nächste Entscheidungstabelle aufgerufen. Um z.B. der Flexibilität "Ausweichstation" zu entsprechen, wird aus der Vorgabe für einen Arbeitsgang eine Entscheidungstabelle mit allen angegebenen Stationen angelegt. Durch das Verbinden dieser Tabellen können dann alle möglichen Arbeitsgangfolgen aufgefunden und auch bewertet werden /33/.

Anhand der Skizzierung eines Algorithmus, der zur Reduzierung einer Produktfamilie in der Tastaturmontagezelle implementiert wurde (vgl. Kap. 7.1), kann aufgezeigt werden, wie jeweils eine konkrete Lösung entwickelt werden kann. Entwurf eines Algorithmus zu dieser Aufgabe:

1. Bestimme über die Daten der zugeordneten AGs alle relevanten RAP-AGs der Fertigungsstufe 1 =: Menge der zu untersuchenden RAP-AGs (M).
2. Wähle einen RAP-AG A aus M.
3. Untersuche alle Folge-RAP-AG, ob sie relevant sind.
Wenn ja, übernehme sie in M.
Wenn nein, streiche sie aus dem RAPL und wenn sie optional sind, untersuche alle Folge-RAP-AG.
Streiche A aus M. Wenn kein relevanter Folge-RAP-AG existiert, dann streiche A aus dem RAPL.
4. Falls M <> 0, gehe zu 2., andernfalls beende die Suche.

Von Fertigungsstufe 1 ausgehend wird versucht, einen Weg durch den Graph zu finden, der nur über relevante Knoten verläuft. Irrelevante Knoten und solche, die nur auf irrelevante Knoten führen, werden entfernt. Es ergibt sich ein Teilgraph der RAPL-Struktur, derart, daß von Fertigungsstufe 1 ausgehend alle Wege durch den Graph vollständig auf relevanten Knoten verlaufen (Es gibt keine Sackgassen!).

6.2.5 Eingliederung in das Ebenenmodell

Mit der Produktfamilie stehen die globale Produktstruktur, der Rahmenarbeitsplan und die zugehörigen Teileklassen aus den Bereichen CAD und CAP z.B. in gemeinsamer Datenbank als Stammdaten zur Verfügung. Ein eintreffender Zellenauftrag wird dann einer bekannten Produktfamilie zugeordnet oder zurückgewiesen. Aus den Teilklassen und den Auftragsdaten wird mit Hilfe eines Algorithmus bzw. Bedienereingaben die intern benötigte Stückliste aufgebaut. Kann der Algorithmus erfolgreich durchgeführt werden, ist gewährleistet, daß der Auftrag zur Produktfamilie gehört; somit erfolgt indirekt eine interne Plausibilitätskontrolle (auch durch Bildschirmmasken erreichbar).

Bei der weiteren Generierung wird die globale Produktstruktur mit den genannten Knotentypen zur auftragsbezogenen Produktstruktur reduziert. Diese beinhaltet den exakten Aufbau (die genaue Montageanordnung der Teile) der zu produzierenden Baugruppe. Mit der Stückliste und der auftragsbezogenen Produktstruktur, die die Materialien und ihre Beziehung untereinander aufzeigt, als Input wird die auftragsneutrale Abarbeitungsfolge im RAPL dann in eine auftragsspezifische Ausprägung umgewandelt (vgl. Bild 47).

Die zweite Reduzierung kann im Ebenenmodell auf Ebene 6 oder 5 eingeordnet werden (Bild 47). Nunmehr werden Informationen über Strukturdaten (z.B. Zuordnungsdaten) und Zustandsdaten (z.B aktuelle Auslastung der Station) benötigt, um zu entscheiden, wie alternative Folgen aufgelöst werden, ob bei parallelen Folgen diese Möglichkeit ausgenutzt wird oder ob einfach nur sequentiell abgearbeitet wird. Erfolgen die Entscheidungen vollständig auf Ebene 6, so werden dann an Ebene 5 die sequentiellen (und parallelen) Arbeitsgänge eines Arbeitsplans weitergegeben, der -sofern der Auftrag aus Teilaufträgen besteht- n-mal vervielfältigt wird. Mit dieser Methode geht jedoch viel an Flexibilität (starre Festlegung für alle Teilaufträge) verloren.

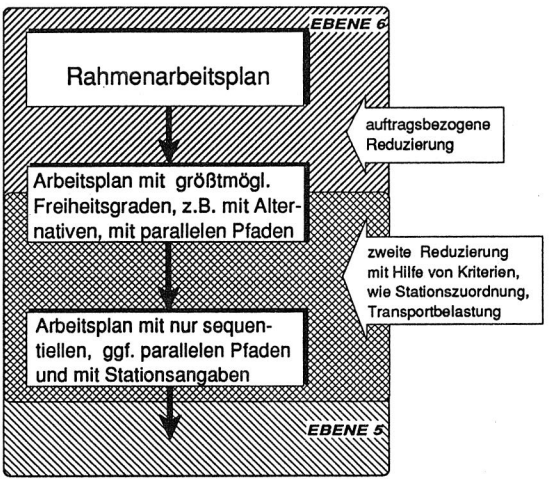


Bild 47: Mögliche Verteilung von Reduzierungsmechanismen

Erfolgt keine Entscheidung auf Ebene 6, so ist der auftragsorientierte APL je Teilauftrag an Ebene 5 weiterzugeben, die dann über die weitere Auflösung der Elemente bei den auftragsorientierten Ausprägungen entscheidet. Jedem parallelen bzw. alternativen Arbeitsgang jeder Ausprägung sind noch alle möglichen Stationen etc. zugewiesen. Bei der zweiten Reduzierung wird jeder Ausprägung einzeln, also n-mal, individuell dynamisch die ausführende Station etc. zugeordnet. Zwischen beiden Extremfällen gibt es verschiedene Variationsmöglichkeiten (Überlappungsbereich in Bild 47; siehe dazu auch Kap. 6.3 und 6.4).

6.3 Ebene 6 - Abwicklung von Zellaufträgen

Ausgehend vom Architekturmodell und den Einflüssen aus Bild 48 können allgemein folgende Aufgaben aufgelistet werden:

- Vorverarbeitung der Auftragsinformation,
- Gesicherte Teilaufträge bilden,
  (Auftrag ist mit verfügbaren Ressourcen ausführbar)
- Betriebsmittelverwaltung, -prüfung und -anforderung,
- Zuordnung und Generierung von Arbeitsplanelementen,
- Einlastungsmechanismen,
- Ausführungsüberwachung.

In bezug auf relevante Daten wie Auftragsbeschreibung, Rahmenarbeitspläne und Zellenlayout sei auf die Beschreibungen in 6.1 und 6.2 verwiesen. Diese Ebene als Schnittstelle zu überlagerten Systemen und als Schnittstelle zur Ablaufsteuerung leitet also durchführbare und freigegebene Teilaufträge an die Ebene 5 zur Bearbeitung weiter und nimmt jeweils teilauftragsbezogen bzw. stationsbezogen Meldungen (z.B. Bestätigung über das Fertigstellen eines Produkts, Störung einer Station) entgegen. Ebenso müssen auch Materialanforderungen von der Stationsebene entgegengenommen und als Bestellung weitergeleitet werden. Die Gesamtüberwachung des Betriebsmittels Montagezelle ist ein weiterer Bestandteil, dazu stellt die Steuerung Kontrollfunktionen, wie zum Beispiel Befehle zur Unterbrechung des Montageprozesses oder zur Initialisierung des Zellenzustands, zur Verfügung.

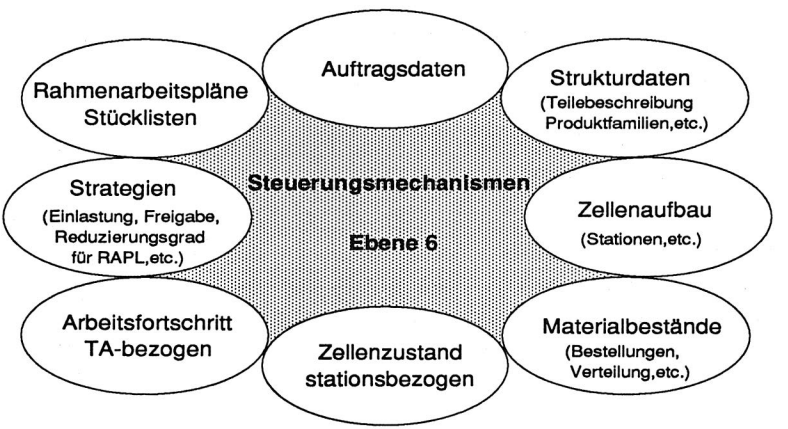


Bild 48: Einflüsse auf Ebene 6

Einerseits wird hier der 'Hardware-Status' des Betriebsmittels Montagezelle kontrolliert, andererseits dient die Protokollierung des Zustands der Aufträge während des Steuerungsprozesses der Initiierung der jeweils nötigen Funktionen bzw. der Fehlerbehandlung im Falle von Systemfehlern (siehe Kap. 4.6). Zu den Zustandsdaten auf der betrachteten Abstraktionsebene werden z.B folgende Elemente gerechnet:

- Zellenstatus
  Er gibt Aufschluß über den momentanen Zustand der Zelle, d.h. es wird vermerkt, ob Elemente der Zelle gerade aktiv bzw. welche Stationen einsatzbereit sind, welche und wieviele Aufträge bearbeitet werden. Weiterhin können diese Daten und vor allem Fehlerzustände für Statistikzwecke, Instandhaltungsmaßnahmen, etc. protokolliert werden.

- Auftragsstatus
  Hier wird der jeweilige Bearbeitungszustand eines Auftrages vermerkt, um geeignete Maßnahmen zur Recovery und Fehlerbehandlung im Falle eines Systemfehlers einzuleiten. Nach Zerlegung eines Auftrags existiert eine Teilauftragsliste mit jeweiliger Zuordnung und entsprechend seines Bearbeitungsfortschritts eine Statuskennung, die z.B. angibt, ob der Teilauftrag bereits eingeplant ist, Werkstücke zur Bearbeitung zugeordnet sind oder wie der Teilauftrag abgeschlossen wurde.

- Auftragsbezogene Daten
  Hierzu gehören die bei der Auftragsabwicklung dynamisch erzeugten Daten zu den einzelnen Montageaufträgen, so z.B eine Liste mit dem Materialbedarf, der reduzierte Arbeitsplan, konkrete Ausführungsstationen mit Durchführungszeiten sowie ermittelte Qualitätsdaten.

6.4 *Ebene 5 - Handling von Arbeitsgängen*

Entsprechend der abstrakten Hierarchieebenen stellt die Einplanung und Verteilung von anliegenden Arbeitsgängen der gesicherten Teilaufträge die zentrale Aufgabe der Ebene 5 dar. Diese Schicht nimmt jeweils die Arbeitsgänge freigegebener Teilaufträge entgegen, sorgt für korrekte Ablauffolgen, die koordinierte Zuweisung zu den Stationen und die Sicherstellung der Verfügbarkeit und Bereitstellung der zugeordneten Werkstücke, Werkzeuge und Programme. Als Ergebnis stellen die Funktionen dieser Schicht jeweils parametrisierte Arbeitsschritte (von der Montageplanung vorgegeben!) für die jeweils eingeplanten und auszuführenden Arbeitsgänge sowie aktuell generierte und auszuführende logische Transporte für die Ebene 4 zur Verfügung und erwarten entsprechende Rückmeldungen (Bild 49).

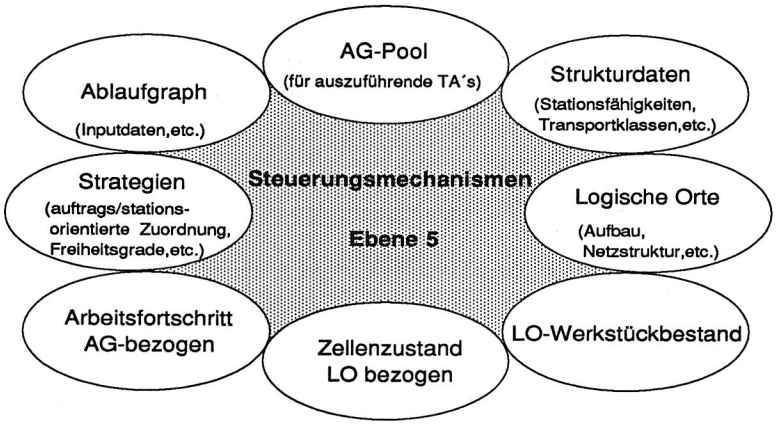


Bild 49: Einflüsse auf Ebene 5

Bei Betrachtung möglicher Zustände in der Arbeitsgangabwicklung (Bild 24) ergeben sich folgende konkrete Teilaufgaben:

- Koordinierung der Arbeitsgänge nach Strukturgraph-Daten,
- Ermittlung aller zur Ausführung anstehenden Arbeitsgänge,
- Zuordnung von Stationen und Werkzeugen zu den Arbeitsgängen innerhalb der gegebenen Freiheitsgrade,
- Optimierung dieser Schritte nach angegebenen Kriterien (Beachtung der Verklemmungserkennung und -vermeidung),
- Zuordnung von Werkstücken zu den logischen Inputs der AGs,
- Einplanung und Planung vorzunehmender Aktivitäten:
  z.B. logische Transporte zum Bereitstellen der Werkstücke,
- Aufbereitung und Freigabe von Arbeitsschrittlisten.

Neben diesen Funktionen existieren auch Hilfsfunktionen zur Unterstützung der Ablaufsteuerung, so unter anderem:

- Funktionen zum Führen eines Materialabbilds für LOs
- Protokollierung der Steuerungsabläufe
- Kontrollfunktionen auf Stationsebene
  z.B. Station anhalten, Station initialisieren, etc.
- Funktionen zur stationsorientierten Nachschubbestellung

Da der RAPL keine Informationen zu physischen Orten und keine Transportarbeitsgänge enthält, kann gerade diese Unabhängigkeit des Montageablaufs von der jeweils aktuellen Materialverteilung in der Zelle wesentlich zur Flexibilisierung der FMZ beitragen. Die Ablaufsteuerung stellt also zustandsbezogen sicher, daß die für die auszuführenden

Arbeitsgänge benötigten Inputteile nach der Stationszuordnung am richtigen Ort zur Verfügung stehen. Die generierten logischen Befehle werden erst auf unteren Ebenen in physische Transporte umgesetzt und entsprechend quittiert. 'Drohende' Konfliktsituationen (z.B. Zyklen von Ausweichtransporten) sollten vom System erkannt und durch entsprechende Algorithmen vermieden werden (vgl. hierzu Kap. 6.4.5).

6.4.1 Zustandsdaten auf Arbeitsgangebene

Sinnvolle Optimierung der Montagearbeitsgänge und die Nutzung der Zellenflexibilität sind ohne die Bereitstellung von aktuellen Informationen gerade bei der Gestaltung dieser Steuerungsebene nicht denkbar, so z.B.

- Arbeitsgangzustandsdaten (AG-Protokoll)
- Stationszustand (Status, Materialbestand, etc.)
- LO-Inhalt

Die Protokollierung der Informationen zu AGs dient einerseits als Basis für die Erfassung und Auswertung von Betriebsdaten der Stationen, andererseits ermöglicht sie die Koordinierung von AGs untereinander sowie von Materialbewegungen und der zugehörigen Freigabe von Arbeitsschritten.

Die erfaßten Betriebsdaten können sowohl dynamisch als auch in speziellen Auswertungsprozessen unabhängig vom Montageprozeß statistisch untersucht werden. Die Auswertung stellt zum einen Informationen für die direkte Optimierung zur Verfügung, zum anderen werden dadurch Anhaltspunkte für die Rekonfiguration einer Zelle zwecks Durchsatzsteigerung oder besserer Einlastungskriterien gewonnen. Außerdem unterstützt das Verfolgen des Montageablaufs für einen Teilauftrag im Fehlerfall die Rekonstruktion der Fehlerursache und die Aktivitäten zur Fortführung bzw. zum Wiederaufnehmen des Betriebs. Über Protokollinformationen zum jeweiligen Bearbeitungszustand werden die anstehenden AGs (Teilmenge aus den RAP-AG-Folgen der gesicherten Teilaufträge) bestimmt. Beispielhafte Elemente eines Protokolls hierfür, für Meldungen an Ebene 6 und für nachträgliche Auswertungen sind:

- Teilauftrag
- RAP-AG-Nummer
- Status
- zugeordnete Station
- zugeordnetes Werkzeug
- zugeordnete Inputteile
- Outputteilnummer und Bezeichnung (z.B. Zuordnungsliste)
- Ausführungszeitinformationen
- Qualitätsmerkmalsdaten

Zudem sind bei der Zuordnung von Arbeitsgängen zu Stationen der notwendige Bestand an Verbrauchsmaterial, die in der Montagestation bereitstehenden Werkzeuge, aktuell im Einsatz befindliche Montagewerkzeuge sowie RC - Programmnummern einer Station zu beachten. Nach der Zuordnung tragen diese Angaben durch einen Abgleich mit den Anforderungen des AGs dazu bei, den erforderlichen Ausgangszustand für den Start eines Arbeitsgangs zu erreichen. Stationszustandsdaten, auch in bezug auf Kontrollfunktionen, können z.B. sein:

- Stationsstatus
- aktueller Arbeitsgang
- aktuelle Werkzeugeinsatzliste
- Materialbestandsliste
- Werkzeugsbestandsliste
- gespeicherte RC - Programme

Die Verwaltung des Inhalts (Paletten, Magazine, Werkstücke) der abstrakten Zellenorte (LOs) unterstützt Steuerungsfunktionen bei der Zuordnung von Werkstücken zu logischen Inputs und bei der Bereitstellung der benötigten Montageteile auf den im Arbeitsplan geforderten Ausgangspunkten. Die Anordnung der Teile auf den LOs und bestimmten Materialträgern bleibt dieser Ebene verborgen. Vielmehr ist wichtig, wieviele der vorhandenen Teile bereits einem Teilauftrag zugeordnet und wieviele noch verfügbar sind, so daß für eine LO-Nummer Informationen wie lagernde Teile (Teilklasse/Teiltyp) mit den Attributen 'verfügbar' oder 'zugeordnet' existieren.

6.4.2 Arbeitsgangauswahl

Die Nutzung der theoretisch möglichen dispositiven Freiheiten hängt zwar entscheidend von der Mächtigkeit realisierter Funktionen dieses Bereichs ab, aber auf Basis ermittelbarer Daten kann hier gezeigt werden, in welcher Form dieser Auswahlprozeß ablaufen kann, welche Randbedingungen zu beachten und welche Kriterien bei der Optimierung und Zuordnung zu berücksichtigen sind.

Die Ausgangssituation zur AG-Auswahl bilden einer oder mehrere zur Abarbeitung anstehende Teilaufträge mit zugehörigen Arbeitsplänen (Freiheitsgrade: Station, Ablauffolge etc.). Ausgehend von Fertigungsstufe 1 muß also ein Weg durch die jeweiligen Strukturgraphen gefunden werden. Dazu existieren die Möglichkeiten der statischen Festlegung vor Beginn oder die dynamische Bestimmung während der Abarbeitung (vgl. Bild 47) unter Angabe eines Planungshorizontes. Nur der zweite Ansatz kann jeweils kurzfristig Zustandsinformationen zur Optimierung des Montageablaufs, d.h. für eine schrittweise Auswahl heranziehen. Über sämtliche Teilaufträge können dann folgende

Schritte für die AG-Auswahl angegeben werden:

- Auswahl anstehender AGs aller Teilaufträge aufgrund des Fortschritts des Montageablaufes (vorher durchzuführende AGs sind abgeschlossen) und Eintrag in die Auswahlliste
- Prioritätensetzung in der Auswahlliste aufgrund unterschiedlicher Kriterien (auftragsabhängig, stationsabhängig, materialabhängig, durchlaufzeitabhängig, ...)
- Reduzierung der anstehenden zu den ausführbaren durch Restriktionen von Material, Stationsauslastung, Stationsbelegung, Transportaufwand, ...)
- Aussuchen und Zuordnung einzelner AGs zu bestimmter Station und Freigabe.

Unabhängig von der angestrebten Optimierung existieren zur Zuordnung zwei grundsätzlich verschiedene Konzepte, die **teilauftragsorientierte** und **stationsorientierte** Auswahl. Dem ersten Ansatz liegt also die Idee zugrunde, vor die Auswahl eines bestimmten Arbeitsgangs die Auswahl eines der aktuellen Teilaufträge zu stellen (Berücksichtigung nur bestimmter Teilaufträge, z.B. angefangene). Erst im nächsten Schritt erfolgt innerhalb des gewählten Teilauftrags die Wahl eines der alternativ anstehenden AGs; hier spielen dann letztlich auch stationsorientierte Kriterien (z.B. welche Station ist verfügbar) eine Rolle. Ein Vorteil dieses Konzepts liegt in der Optimierung des Montageablaufs hinsichtlich Kriterien, die sich an den Aufträgen orientieren (z.B. Minimierung der Verweilzeit), vgl. Bild 50.

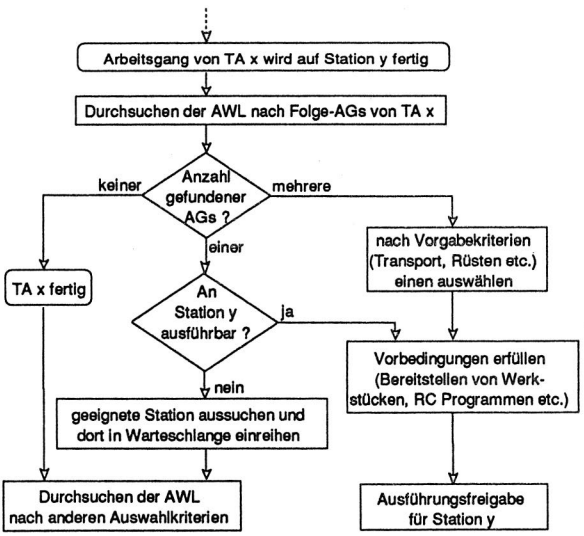


Bild 50: Auftragsorientierte Auswahl mit Stationskriterien

Bei einer rein auftragsorientierten Auswahl kann es vorkommen, daß zwar AGs der Auswahlliste (AWL) an bereiten Stationen ausgeführt werden könnten, diese jedoch nicht zu den vorzuziehenden Aufträgen gehören. Damit bliebe eine Station unbenutzt, die Auslastung würde sich verringern.

Im Gegensatz dazu wird die stationsorientierte Auswahl dadurch ausgelöst, daß Stationen frei und bereit sind, typischerweise dann, wenn ein anderer AG abgeschlossen wurde. Unabhängig von der Zugehörigkeit zu Teilaufträgen sucht das System den geeignetsten AG für die Station aus. Hier wird also solange nach ausführbaren AGs in der AWL gesucht, bis keine Station (auch ersetzende) mehr bereit ist.

Die Suche kann beliebig kompliziert gestaltet werden, um das Ziel der optimalen Auslastung der Stationen und der Minimierung der Rüstzeiten (geringe Wartezeiten) zu erreichen. Diese Vorgehensweise erleichtert die Optimierung in bezug auf stationsorientierte Kriterien wie Stationsauslastung, Minimierung des Transportaufwands oder Durchsatzmaximierung. Das Hauptproblem besteht schließlich darin, daß - abhängig vom Suchaufwand (spezielle rechenzeitintensive Algorithmen) - häufig Ersatzstationen gewählt werden und möglicherweise immer neue Teilaufträge begonnen werden, wodurch die Zahl der Zwischenprodukte ansteigt.

Je nach Optimierungsschwerpunkt kann ein Konzept bevorzugt werden und ein komplexes kombiniertes Auswahlverfahren durch höhere Gewichtung prägen. Hierdurch werden jeweilige Flexibilitätskriterien (vgl. Kap. 3.1) berücksichtigt, so daß man in konkreten Anwendungen bessere Suboptima erreicht.

Auswahlliste (AWL)

Beiden Ansätzen gemeinsam ist, daß nach jeder Auswahl und Zuweisung eine Aktualisierung der Menge der anstehenden AGs vorgenommen werden muß, z.B Löschung eventuell vorhandener Alternativen zum gewählten AG. Im Prinzip reicht eine Auflistung, die aufgrund der jeweiligen Abarbeitungsstufen die zur Abarbeitung anstehenden RAP-AGs enthält. Nach der Freigabe eines Teilauftrages wird der Strukturgraph -ausgehend von der Wurzel- gelesen und der AG des ersten Knotens bzw. die AGs (bei Alternative und Parallelität) werden mit Teilauftragsidentität in die Auswahlliste eingetragen.

Nach Auswahl von AGs aus der AWL werden diese zur Ausführung freigegeben und in der AWL gekennzeichnet sowie Alternativen entfernt. Gleichzeitig können aus den vorliegenden Strukturgraphen die Folgeknoten ermittelt und eingetragen werden. Sobald

ein AG als "fertig" gemeldet wird, wird er aus der AWL gelöscht und evt. in einer Protokolldatei weitergeführt. Gibt es keinen Folgeknoten mehr, so gilt der Teilauftrag als "fertig".

Auswahlbedingungen (AWB)

Ein AG ist aus der Sicht der Ablaufsteuerung nur dann ausführbar, wenn die folgenden fünf Bedingungen erfüllt sind bzw. im Planungshorizont erfüllt werden können:

1) Vorher durchzuführende Arbeitsgänge müssen eingeplant bzw. abgeschlossen sein. (Bedingung zur Aufnahme in AWL)
2) Eine potentielle Station muß bereit sein. (Bedingung für Stationszuweisung)
3) Alle benötigten Teile müssen am Platz sein. (Bedingung für den Start des AG)
4) Für die entstehenden Teile muß freier Platz zur Aufnahme vorhanden sein. (Bedingung für den Start des AG)
5) Werkzeuge und RC-Programme müssen verfügbar sein. (Bedingung für Stationszuweisung und Start)

Die erste Bedingung wird dadurch erfüllt, daß ohnehin nur anstehende Arbeitsgänge in der AWL stehen. Die Bedingung 2) ist eine elementare Voraussetzung für eine Auswahl und kann durch Meldungen ('frei') bzw. Belastungskriterien erreicht werden. Eine potentielle Station zur Ausführung ist nämlich genau dann vorhanden, falls die Bedingungen 3), 4) und 5) erfüllt sind. Potentielle Stationen zu einem AG sind z.B. dem zugeordneten LAG (Folge von logischen Arbeitsschritten mit Stationsangaben) zu entnehmen.

Beachte: Je mehr Auswahlmöglichkeiten sich ergeben, desto flexibler wird die Montagezelle, aber umso wahrscheinlicher werden andererseits ein Ansteigen der Lagerkosten, der benötigten Teile bei den verschiedenen Stationen, und des gesamten Transportaufwandes sowie der Transportkosten.

Die Bedingung 3) betrifft nicht nur Teile in der Station, sondern auch Teile (evt. montierte Baugruppen), die dorthin gebracht werden müssen (Einleitung von Transporten). Damit gilt: Ist Bedingung 3) erfüllt, dann ist auch 1) erfüllt, d.h. aus 3) folgt 1), aber nicht notwendig umgekehrt.

Den logischen Inputteilen der AGs sind geeignete Werkstücke zuzuordnen, die innerhalb des Teilauftrags eindeutig durchnumeriert werden. Bekanntlich repräsentiert jeder RAP-AG genau einen AG, dem durch den Verweis auf einen Knoten in der Erzeugnisstruktur eine Menge von Inputteilen zugeordnet wird. Werkstücke in der Zelle sind dann als Input geeignet, wenn ihre Bezeichnung bestehend aus Teilklasse und Teiltyp mit der im jewei-

ligen Knoten der Erzeugnisstruktur gespeicherten Bezeichnung übereinstimmt. Besteht bezüglich eines Inputteils die Auswahlmöglichkeit aus mehreren real vorhandenen Montageteilen, so kann entsprechend vorgegebener Optimierungskriterien (z.B. Transportaufwandsabschätzung) ausgesucht werden.

Die Auswahl kann aber auch allein nach protokollierten Qualitätsdaten für ein Werkstück getroffen werden. Die Informationen zur Ermittlung des Bereitstellungsortes von Werkstücken enthält der jeweilige Inputsatz des AG, zur Umsetzung dieser Information vgl. Kap. 6.1. Die Zuordnungen des Verbrauchsmaterials erfolgt erst in der Ebene 4, d.h in der Handlungs-/Verwaltungsebene der Stationen.

Die Bedingung 4) gewinnt insbesondere bei der überlappten Ausführung von Teilaufträgen unterschiedlicher Produktfamilien an Bedeutung. Die Flexibilität der Montagezelle wird dadurch erhöht, daß neu entstandene Teile sowohl abtransportiert als auch zwischengelagert werden können. Eine Zwischenlagerung ist vorallem bei der Weiterverwendung in der Station als geeignet anzusehen. Als Problem ist jedoch der Platzbedarf und das Platzangebot in der Station zu beachten.

Die Bedingung 5) wird zellenbezogen zwar bereits von der Ebene 6 zugesichert, aber zur Kontrolle bzw. zum Austausch von Werkstücken, Verbrauchsmaterial, Werkzeugen oder RC-Programmen wird diese Bedingung auch hier berücksichtigt. Ein Wechsel des logischen Werkzeugs kann sowohl innerhalb des Arbeitsgangs (Aktivität ist Bestandteil der Ebene 4) durchgeführt als auch als eigener Arbeitsgang betrachtet werden.

Zusammenfassend bleibt festzuhalten, daß die Ablaufsteuerung bei der Auswahl von Arbeitsgängen aus der Auswahlliste die Bedingungen 2) bis 5) auf Erfüllbarkeit hin prüfen muß. Zusätzlich erstrebenswert erscheint die Bestimmung des benötigten Aufwands (zeitlich, wirtschaftlich, etc.), um diese Werte als Auswahlkriterium mit einzubeziehen.

Optimierungskriterien

Werden in der Planung (unabhängig von der Wahl des Planungszeitraums) für einen zukünftigen Zeitpunkt mehrere Alternativen der Fortsetzung erkannt, so besteht auf Zellenebene -abhängig von der zu optimierenden Größe- die Möglichkeit, die "beste" Alternative auszuwählen. Zur Fortsetzung der Teilaufträge sind folgende Optimierungskriterien denkbar:

1) Durchsatzmaximierung

Verbunden mit hoher Stationsauslastung wird versucht, die anfallenden Montageoperationen möglichst parallel auszuführen. Während der Ausführung eines AG können dabei bereits die Voraussetzungen hinsichtlich Teile- und Werkzeugbereitstellung für einen weiteren AG erfüllt werden.

2) Minimierung der Auftrags- und Teilauftragsverweilzeit

Die Zugehörigkeit zu Teilaufträgen und Aufträgen bestimmt die Auswahl der AG, d.h. alle AG eines Teilauftrags werden möglichst hintereinander oder auch parallel ausgeführt.

3) Maximierung der Stationsauslastung

Sobald eine Station einen AG abschließt, muß sie mit der Ausführung eines neuen AG beauftragt werden. Um den Zeitverlust, der durch den Auswahlprozeß für den neuen AG entsteht, zu vermeiden, können im voraus bereits mehrere wartende AGs bestimmt werden. Nachteil dabei ist allerdings, daß die Auswahl nicht auf den aktuellsten Zustandsdaten aufsetzt.

4) Minimierung des Transportaufwands

Durch die bevorzugte Zuordnung von Arbeitsgängen zu Stationen, bei denen die benötigten Montageteile bereits vorliegen oder deren Bereitstellungsort am einfachsten zu erreichen ist, wird die Belastung des Materialflußsystems der Zelle minimiert. Bei allzu konsequenter Verfolgung dieses Ziels besteht jedoch die Gefahr, daß gewisse Stationen weniger ausgelastet werden, bzw. daß die Verweildauer der Teilaufträge in unvertretbarer Weise zunimmt.

Sicherlich muß jeweils nach den Bedürfnissen der Anwendung ein Kompromiß zwischen Optimierungsaspekten gefunden werden. Vorallem Protokolldaten sollten von Zeit zu Zeit einer statistischen Auswertung unterzogen werden, um dann die Optimierung nach neuen Erkenntnissen ausrichten zu können. Denn weder ist es möglich, alle Größen gleichzeitig auf einen optimalen Wert zu bringen, noch ist es praktisch vertretbar, eine Größe beliebig auf Kosten der anderen zu gewichten. Für eine "richtige" Abwägung sind besonders auch die Kriterien für eine vorzeitige Neuplanung zu beachten. Prinzipiell sollten Planungsvorgaben dann abgebrochen werden, wenn sie entweder nicht mehr weiter verfolgt werden können oder wenn die ihnen zugrunde liegenden Annahmen über den Zustand der FMZ nicht mehr zutreffen. Denkbare sinnvolle Kriterien für eine vorzeitige Neuplanung sind:

- Das Fehlschlagen eines Arbeitsgangs oder eines Transportes
- Der Ausfall oder die erneute Verfügbarkeit einer Maschine
- Das Eintreffen eines neuen Fertigungsauftrages
- Das Eintreffen von Werkstücken, Verbrauchsmaterial etc.

6.4.3 Modularisierungsaspekte am Beispiel der Koordinierung

Gerade bei der Arbeitsgangabwicklung sind Koordinierungs- und Synchronisationsmechanismen unverzichtbar und eignen sich deshalb für ein beispielhaftes Aufzeigen von Aspekten und Vorteilen einer Modularisierung. Unter Koordinierung wird das "logische Zusammenwirken von Teilen" und die dazugehörigen Maßnahmen, unter Synchronisation dagegen das Abstimmen verschiedener Vorgänge auf Gleichzeitigkeit verstanden /48/. Betrachtet man die Tätigkeiten auf dieser Ebene als das Vergeben von Betriebsmitteln (z.B Fügegerät, Material), so ist jede Vergabe in sinnvoller Weise zu koordinieren. Zu den Betriebsmitteln wird auch folgendes gerechnet:

- die Fähigkeit einer Station, einen Arbeitsgang auszuführen,
- das Vorhandensein eines Teiles (Werkstücke, Baugruppen...) an einem bestimmten Ort innerhalb der FMZ,
- freie Lagerkapazität an bestimmten Orten der FMZ,
- Fähigkeiten des Transportsystems, Teile zu bewegen,
- Fähigkeiten des Handhabungssystems, Teile bereitzustellen.

So ist z.B. bei der Zuweisung einer Station an einen AG und beim Start dieses AGs die Abhängigkeit mit allen notwendigen Betriebsmitteln (Übergabeplatz, Werkzeug, Werkstelle,...) zu untersuchen und aufzulösen. Die Koordinierung eines AGs mit einem anderen AG kann der Herstellung einer räumlichen (stationsbezogenen) oder zeitlichen (chronologischen) Beziehung (Bild 51) der Arbeitsgänge untereinander dienen.

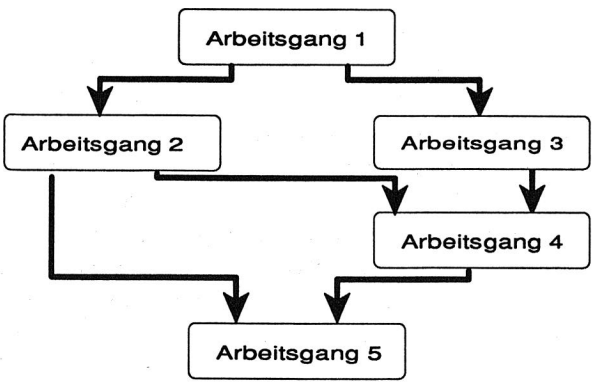


Bild 51: Chronologische Beziehungen zwischen zwei AGs

Zusammengefaßt ist die chronologische Reihenfolge für Koordinationen wegen der Bearbeitungsfolgen, wegen Vorhandensein von Teilen (an einem bestimmten Ort) und

wegen Blockierung in der Zelle notwendig. Letzteres ist zu beachten, falls bestimmte AGs eine Station nur für bestimmte Fortsetzungsarbeitsgänge freigeben, d.h. es muß sichergestellt sein, daß auch die 'deblockierenden AGs' durchführbar sind.

Weitere Koordinierungsaufgaben betreffen freie Pufferkapazitäten, um eine Blockierung der Station zu vermeiden. Die Koordinierung von AGs mit der Transportkapazität des Transportsystems (Durchsatz und gleichzeitige Ausführung) in der Zelle muß verhindern, daß Teile während der Bearbeitung bewegt werden und daß sich Transporte gegenseitig blockieren.

Die Synchronisation von Vorgängen (z.B. AG-Start auf einer Station oder Start eines Transportes) ist in folgenden Varianten denkbar:

- Ein Vorgang muß bei Eintreten eines Ereignisses gestartet werden (absolute Synchronisation).
- Ein Vorgang muß mindestens/genau/höchstens eine bestimmte Zeit vor/nach einem Ereignis gestartet werden (relative Synchronisation).

Die Synchronisation ist nur zwischen zwei "Vorgängen" oder zwischen einem "Vorgang" und einem "Ereignis" möglich. In der Praxis ist die Synchronisation zweier AGs nötig, wenn innerhalb des Montageprozesses gleichzeitig Arbeiten am selben Werkstück verrichtet werden.

Allgemein haben sich in flexiblen Montagezellen mit mehreren Stationen die folgenden Koordinierungs- und Synchronisations-Aufgaben herauskristallisiert:

- Koordinierung zwischen zwei AGs bezüglich der Stationen,
- Koordinierung eines AG mit dem Vorhandensein eines Teiles an einem bestimmten Ort der FMZ,
- Koordinierung eines AG mit freier Lagerkapazität,
- Koordinierung von Transporten untereinander entsprechend den Erfordernissen der Transportsysteme,
- Koordinierung von Transporten mit laufenden AGs,
- Synchronisation des Starts zweier Arbeitsgänge.

In der jeweiligen Steuerungssoftware sollte also eine dynamische, aber blockierungsfreie Zuordnung von Arbeitsgängen sowie die Generierung und Verteilung von Transportanweisungen möglich sein. Unabhängig jeweiliger Mechanismen zum Erkennen und zur Erfüllung der Anforderungen werden zunächst obengenannte Aufgaben jeweils als eigene Module betrachtet. So können dann bei der Arbeitsgangauswahl (ein Hauptbestandteil der Arbeitsgangabwicklung) unterschiedliche Auswahlverfahren (teilauftrags-

bzw. stationsorientiert etc.) zugelassen werden, einerseits wegen der Berücksichtigung jeweils notwendiger Koordinierungs- und Synchronisationsmechanismen aufgrund unterschiedlicher Zellenausprägungen und andererseits wegen der geforderten Durchsetzung unterschiedlicher strategischer Ziele (vgl. Optimierungskriterien).

Verklemmungen können immer dann auftreten, wenn Betriebsmittel (z.B. Werkzeuge, Transportkapazitäten, Kapazität der log. Orte, Stationsfähigkeiten), die nur in begrenztem Umfang zur Verfügung stehen, an konkurrierende Verbraucher zu vergeben sind. In Ebene 5 einer FMZ sind z.B. die Verklemmungen 'Ein Transport (LOx - LOy) kann nicht ausgeführt werden' und 'Ein logischer Ort kann kein Teil mehr aufnehmen' denkbar und zu vermeiden. Die erste Art kann durch Auswertung von Zustandsdaten und Strukturdaten (Aufbau und Verbindung von LOs) gelöst werden. Die zweite Art von Verklemmungen kann zuverlässig vermieden werden, wenn von jeder Aktion, die in den Plan aufgenommen wird, genau bekannt ist, wieviel Kapazität auf welchem LO sie konsumiert oder freisetzt. Für jeden Schritt des Planes kann so die Verklemmungsfreiheit garantiert werden.

Es zeigt sich also, daß Komponenten, d.h. auch implementierte Softwarebausteine, mehrfach in verschiedenen, nicht vollständig vorhersehbaren Kontexten verwendet werden können, wenn zunächst allgemein Aufgaben der Komponenten spezifiziert werden. Sie sollen so wenig wie möglich von Bedingungen abhängen, die mit großer Wahrscheinlichkeit nur bei einer konkreten Verwendung zutreffen. Es bedarf also wiederum Mittel (z.B. ADT, PDT), mit denen gefundene Aufgaben ausreichend abstrakt beschreibbar sind und mit denen auch allgemeine und spezielle Teile (konkrete Anwendungserweiterungen und konkrete Werte für parametrisierte Komponenten) getrennt werden können, sodaß sich jeweils unterschiedliche Instanzen ergeben.

6.5 Ebene 4 - Abarbeitung von Arbeitsschritten

Im Mittelpunkt der Steuerungsfunktionen dieser Ebene steht die Generierung der Parameterwerte eines logischen Arbeitsschrittes und seine Umsetzung in eine Folge stationsbezogener Montageaktionen. Bisher wird an dieser Stelle der Arbeitsplanhierarchie dem(n) übergebenen Arbeitsschritt(en) meist ein festes RC-Programm (ohne Parametrisierungsmöglichkeiten) zugeordnet. Ziel des hier verfolgten Ansatzes ist es, aus einem 'stationsunabhängigen' logischen Befehl mit Unterstützung durch ein Datenmodell eine Reihe von Aktionsanweisungen, bezogen auf physisch vorhandene Montagestationen, zu generieren (Bild 52). Den übergeordneten Abstraktionsstufen der Steuerungshierarchie

wird somit eine stationsunabhängige Schnittstelle zur Verfügung gestellt, so daß die dort relevanten Arbeitsplanelemente von den tatsächlich vorhandenen Hardwarestrukturen unabhängig bleiben.

Generierung der Arbeitsschrittparameter

Die Inputteile für den Arbeitsgang sind auf die Basis- und Fügeteilparameter in der jeweils zum AG gehörigen Arbeitsschrittliste abzubilden. Hierbei wird jedem LAS eine Teilmenge der Inputsätze des AG zugeordnet. Auf dieser Teilmenge definiere man eine Ordnung, die die Reihenfolge der Zuweisung der Inputs zu den AS angibt. So läßt sich ausgehend vom ersten Inputsatz bei der Auflösung des AG in eine AS-Liste jedem AS ein Knoten in der Erzeugnisstruktur (Teilbezeichnung - Klasse/Typ) zuordnen. Der Inputsatz gibt jeweils die logischen Orte an, wo die Teile bereitgestellt werden. Bei Verbrauchsmaterial wird vorausgesetzt, daß es in der Peripherie der Station vorhanden ist; Werkstücke liegen aufgrund fester Zuordnungen durch Ebene 5 und generierter logischer Transportbefehle (AG-Transport) bereits auf dem geforderten LO bereit.

Ebenso ist den AG-Inputs eine Liste mit den logischen Fügeorten (basisteilbezogen) für den LAS-Satz zugeordnet. Diese logischen Fügeorte verweisen dann z.B. auf Grundlage eines werkstückorientierten Koordinatensystems (Produktmodell !) auf reale Orte des Basisteils. Die Fügeortdaten werden dann im Weltkoordinatensystem des IR aufgrund der Lage und Orientierung der bereitgestellten Basisteile ermittelt und dienen auch als Grundlage für die Bestimmung von Koordinaten der Handhabungs- und der Fügebewegung selbst.

Um einem gegebenen LO auch einen physischen Ort in der Station zuordnen zu können, muß eine exakte Beschreibung der Stationsperipherie existieren. Lösungen (vgl. 7.5.2) hierfür können sowohl Vektor- und Matrixstrukturen (vgl. Realisierung der integrierten FMZ) als auch eine rekursive Gliederung der Peripherie in Elementarstrukturen und zusammengesetzte Strukturen (vgl. Realisierung der Tastaturmontagezellensoftware) sein. Zusammengesetzte Strukturen dienen der Beschreibung von komplexen Objekten wie Paletten oder ausgezeichneten Orten wie Halte- oder Bearbeitungsplätzen, die selbst wieder Unterstrukturen enthalten können.

Für jede zusammengesetzte Struktur ist innerhalb der jeweiligen Oberstruktur ein Koordinatensystem definiert, auf das sich die Positionsangaben der Unterstrukturen beziehen. Bei den Elementarstrukturen handelt es sich um einzelne Montageteile oder Material- / Werkzeugmagazine. Für sie ist jeweils ein Positionswert bezogen auf die umgebende

Struktur enthalten. Beim Ermitteln der Position von Fügeteil und Basisteil wird das Peripherieabbild nach den entsprechenden Teilen durchsucht. Jedem logischen Ort zu der Station, der Bereitstellungsort für ein Teil ist, ist dabei eine Struktur zugeordnet, die den Ausgangspunkt für die Suche bildet.

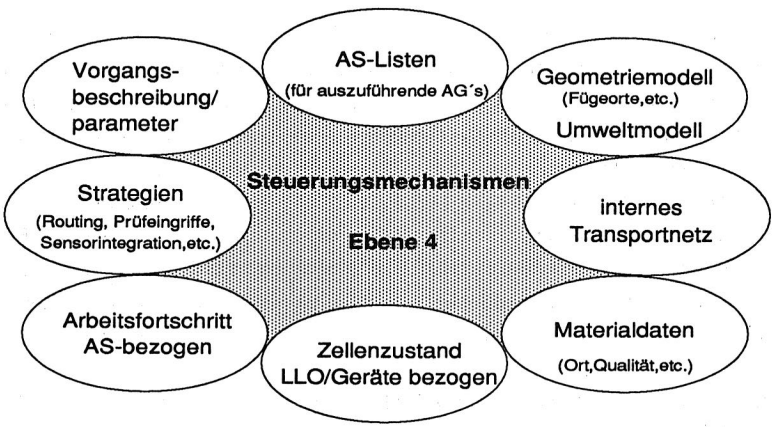


Bild 52: Einflüsse auf Ebene 4

Zerlegen eines Arbeitsschrittes in Operationen

Nach Erzeugung der 'Pick and Place' Werte geht es beim technologischen Aspekt darum, - je nach Ausreifung des Geometriemodells bzw. der Freiheiten der technologischen Angaben im LAS - die AS in Teilaktionen zu zerlegen. Diese stellen dann auf der Ebene 3 die Grundlage für die Generierung von Robotersteuerbefehlen dar.

Zum Zwecke einer anschaulicheren Beschreibung des Vorgangs beschränken sich die Ausführungen auf die beispielhafte Betrachtung eines bestimmten Fügeverfahrens. Angesichts der Vielzahl der Fügeverfahren und deren Ausprägungen für Anwendungen sowie der Häufigkeit in der Praxis wurde das Verfahren 'Fügen durch Einlegen' gewählt. Bei der Übertragung der erarbeiteten Lösungsansätze sind verfahrensspezifische Gesichtspunkte entsprechend zu modifizieren bzw. zu ergänzen.

Die zugehörige Verfahrensgruppe 'Zusammenlegen' charakterisiert Spur /104/ folgendermaßen:

- Die Wirkungsweise der erzeugten elementaren Verbindungen ist grundsätzlich formschlüssig. Die räumliche Anordnung der gefügten Teile kann aber außer durch Formschluß durch einen Kraftschluß, etwa durch Schwerkraft, Reibkraft oder Kraft durch elastische Bauteilverformung gesichert werden.
- Die gefügten Teile berühren sich unmittelbar ohne Hilfsfügeteile und Hilfsstoffe.
- Durch den Fügevorgang werden die Fügeteile nicht plastisch verformt.

DIN 8593 unterscheidet neben 'Einlegen' fünf weitere Untergruppen. Die Definition von 'Einlegen' ist bei Spur /104/ nachzulesen. Die Betrachtung des gesamten Fügevorgangs läßt eine Einteilung in drei Phasen sinnvoll erscheinen:

1. Aufnehmen des Fügeteils
2. Handhaben
3. Fügen

Aufnehmen des Fügeteils

Nach der Beantwortung der Frage 'Wo?' zum Zwecke der Aufnahme des Fügeteils durch die vorher aufgezeigten Mechanismen, ist noch das 'Wie?' zu klären. Grundsätzlich ist dazu anzumerken, daß die Fixierung des Teils so erfolgen soll, daß für den späteren Einlegevorgang keine Änderung des Greifpunkts und auch keine Neuorientierung des Greifers zum Teil notwendig wird. Dazu müssen aus den Geometriedaten des Teils und der Beschreibung der Fügeendlage von Basisteil und Fügeteil geeignete Greifpunkte und eine passende Orientierung des Werkzeugs zum Teil ermittelt werden.

Erfolgt die Bereitstellung aus einem Magazin, so ist zu überprüfen, ob die Lage des Teils im Magazin eine derartige Entnahme zuläßt. Analog muß für Einzelteile auch deren eigene Lage am Bereitstellungsort berücksichtigt werden. Es ist Aufgabe der Steuerung - auch mit Hilfe von Informationen aus der Greifplanung - zu erkennen, ob ein geeignetes Greifen möglich ist oder ob Umgreifen notwendig wird.

Zusätzlich soll bei der Magazinentnahme die Aufnahmebewegung in Abhängigkeit vom Typ des Magazins ausgeführt werden. Insbesondere sind dabei der Weg und die Geschwindigkeit der Aufnahmebewegung betroffen. Soweit das Greifwerkzeug eine derartige Parametrisierung zuläßt, soll die Fixierung des Fügeteils mit einer, von Gewicht und Material des Teils abhängigen Kraft erfolgen.

Handhaben

Mit dieser zweiten Phase sind folgende Aufgaben verbunden:

- Bestimmung der Fügeausgangsposition und -orientierung des Fügeteils zum Basisteil
- Transport des Fügeteils vom Bereitstellungsort zum Fügeort
- Einnehmen der ermittelten Orientierung zum Basisteil
- Falls notwendig, Sicherstellen der geeigneten Beziehung Fügeteil - Greifer durch Umgreifen

Ähnlich der Bestimmung von Greifpunkten und Greiforientierung in der Aufnahmephase muß für den Einlegevorgang die Ausgangsposition und -orientierung des Fügeteils zum

Basisteil berechnet werden. Durch die Einbeziehung der Werkzeugkoordinaten bezüglich des Fügeteils ergibt sich daraus auch der Anfangspunkt der Fügebewegung in Gerätekoordinaten, der zugleich Endpunkt für die Handhabungsbewegung des Fügeteils ist. Analog läßt sich über die Orientierung des Greifers zum Fügeteil auch die erforderliche Bewegung des Werkzeugs zur Einnahme der Fügeausgangsorientierung steuern.

Zur Optimierung des Bewegungsablaufs sollen die Bewegungen zum Erreichen des Fügeorts und zur Einnahme der Fügeausgangslage soweit wie möglich parallel ausgeführt werden. Dies ist jedoch erst Aufgabe der untergeordneten Ebene, die die Erzeugung von RC-Befehlen für die Aktionen aus Ebene 4 übernimmt (auch in bezug auf eine Kollisionsbetrachtung). Soweit es die Möglichkeiten des Handhabungsgerätes zulassen, müssen auch Bewegungsgeschwindigkeit und Beschleunigung in Abhängigkeit vom bewegten Teil und der entsprechenden Geräteparameter (Grenzwertberücksichtigung) bestimmt werden.

Fügen

Zeitlicher Anfangspunkt und Endpunkt von Handhaben und Fügen können dabei durchaus unterschiedlich sein. Über den Weg, auf dem beim Einlegen verfahren wird, ergibt sich aber für die Steuerung die Möglichkeit die relativen Geschwindigkeiten sowie Anfangs- und Endpunkte der Teilbewegungen (Übergangsanordnung) zu ermitteln. Als Übergangsanordnung kann diejenige räumliche Anordnung des TCP vom Handhabungsgerät angesehen werden, ab der Parameter der Technologieoperation Einfluß auf die Bewegung nehmen (Verlangsamen, engere Toleranzen, etc.). Aus der Bewegung des Teils läßt sich dann auf Ebene 3 unter Berücksichtigung der Greifer-Teil Beziehung die notwendige Bewegung des Handhabungsgerätes bzw. des Greifwerkzeugs berechnen.

Es ist darauf zu achten, daß die Teile während des Einlegens unter Umständen Belastungen ausgesetzt sind, die beim Generieren der Bewegungsbefehle zu berücksichtigen sind. Eine dynamische Simulation des Einlegevorgangs kann über Grenzwertabschätzungen für die auftretenden Belastungen die optimalen Parameter für Geschwindigkeiten und erlaubte Kräfte liefern. Im besonderen müssen auch für das Lösen des Greifers Geschwindigkeit und Weg errechnet werden. Die Einbeziehung von Toleranzen bezüglich Positioniergenauigkeit, Geometrie der Teile und erlaubte Belastung gibt darüberhinaus Aufschluß, welche Sensorinformationen für die Durchführung der Aktionen nötig sind.

Es kann so bestimmt werden, ob die Sensorik nur die Ausgangssituation vor Beginn des Einlegens zu überprüfen hat oder ob die erhaltenen Sensorinformationen dynamisch während des Vorgangs auszuwerten sind. Die eigentliche Auswertung der Sensordaten

und die Erzeugung eventueller Korrekturbefehle erfolgt dann erst auf Ebene 3, der Schnittstelle zu den Robotersteuerungen.

Bei Classe und Scholz /12/, die die Bedeutung der verfahrensbeschreibenden Parameter in der Montageautomatisierung untersuchten, werden unterschiedlichste Verfahrensparameter angegeben. Derartige Hilfsfunktionen, zu denen beispielsweise auch die Verwendung von Gleitmitteln und die Regelung des Drahtvorschubs beim Schweißen gehören, sind für das Fügeverfahren 'Einlegen' in der Praxis weitgehend irrelevant. Bei modularen Systemen können für andere Verfahren solche anwendungsbezogenen Parameter ergänzt und zusätzliche Steuerungsfunktionen integriert werden.

Problempunkte bei der Realisierung

Bei den Überlegungen der vorhergehenden Abschnitte handelt es sich um konzeptionelle Überlegungen zu einer Flexibilisierung von Montagesteuerungen auf der Ebene der Arbeitsschrittausführung. Inwieweit Probleme mit der Realisierbarkeit eines solchen 'impliziten' Konzepts auftreten können, werden im Rahmen anderer Arbeiten näher untersucht /36/. Es sollen hier aber in bezug auf eine Softwaregestaltung einige Punkte, die unter diesem Aspekt zu beachten und noch genauer auszuarbeiten sind, herausgehoben werden:

1. Die Bereitstellung der Geometriedaten durch relationale Geometriemodelle unterstützen und erleichtern unter den gegebenen Randbedingungen den Aufbau von Softwaresystemen. Zudem muß schon bei der Konstruktion eines Produkts und seiner Einzelteile dafür gesorgt werden, daß die für die Steuerung nötigen Geometriedaten (z.B. Fügeendlage) zur Verfügung gestellt werden.

2. Die Komplexität der Algorithmen zur Errechnung der geometrieabhängigen Steuerparameter erfordert neben einer Einsetzbarkeitsstudie für die relativ gerätenahe Steuerungsebene 4 die Integration derartiger Algorithmen in bestehende Systeme, um den Aufwand abschätzen zu können und in vertretbaren Dimensionen zu halten.

3. Für die Simulation des Fügevorgangs muß spezifiziert werden, welche Daten in die Simulation eingehen, wie die Auswertung aussehen soll und welche Werte man durch die Simulation gewinnen will. Die Dynamik des Fügeprozesses verlangt auch hier kurze Reaktionszeiten!

7 Anwendungen und Realisierungserfahrungen

Die beschriebene Systemphilosophie zur Entwicklung von Steuerungssoftware wurde für Montagezellen in einer Modellfabrik eingesetzt. Auf Basis dieses Konzeptes wurde weiterhin durch Untersuchungen und Implementierungen aufgezeigt, inwieweit sich relationale Datenmodelle sowohl zur Beschreibung der abstrakten Datenzusammenhänge als auch bei der Datenhaltung selbst eignen. Anhand parametrisierbarer Rahmenarbeitspläne wurde die Möglichkeit geschaffen, durch Auswertung von Zustandsdaten und durch Generierungsmechanismen innerhalb der Zelle die Systemflexibilität zu erhöhen.

7.1 Doppelroboterzelle

Eine erste Anwendung für diese Art Zellensteuerung (das Softwaresystem MOST90 auf einem Industrie-PC) bestätigte die Realisierbarkeit solcher Konzeptionen. Der Montagezelle (Bild 53) obliegt die Aufgabe, mittels zweier Roboter aufgrund einer auftragsspezifischen Anforderung die automatische Montage von mehreren Varianten und Typen einer Vormontagebaugruppe (Tastaturgrundkörper mit unterschiedlichen Tastenanordnungen) und das Einsetzen dieser und anderer Werkstückvarianten (z.B Elektronikplatinen) in ein Gehäuse durchzuführen.

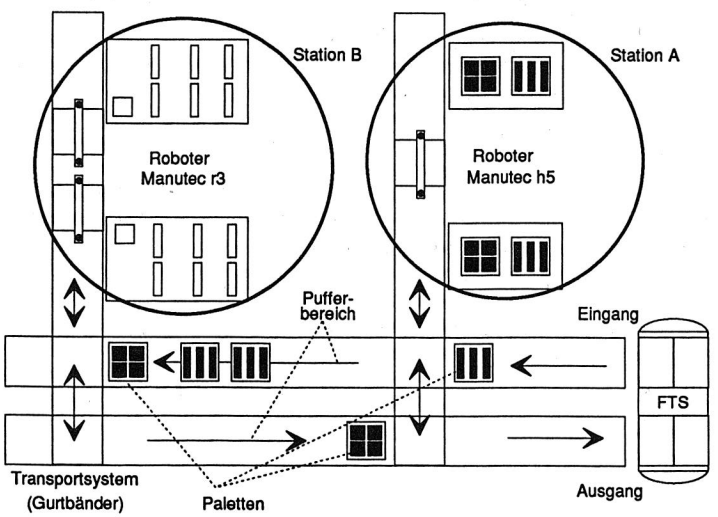


Bild 53: Layout der Tastaturmontagezelle

Der Materialfluß zwischen den Robotern wird über ein Transportsystem (Doppelgurtband) ermöglicht, das gleichzeitig für den An- und Abtransport einzelner Bauteile und Verbrauchsmaterial zu einem FTS zuständig ist. Am Eingang der Zelle befindet sich eine Lese- und Beschriftungsstation (LBS), die die Identifikation der ankommenden Materialpaletten durchführt. Die Koordination der verschiedenen Stationen erfolgt ausschließlich durch den Zellenrechner. Als Hardware für den Zellenrechner stand ein Sicomp PC 16-20, ein IBM AT-kompatibler Rechner, mit 70MB Festplatte und 2.5 MB Hauptspeicher zur Verfügung. Über eine DCP/MUX-Kommunikationsbaugruppe ist er mit zwei Robotersteuerungen, einer speicherprogrammierbaren Steuerung (SPS) für das Transportsystem und mit einem Anschaltmodul für die mobile Lese- und Beschriftungsstation (LBS) verbunden. Die Integration eines Kommunikationsprozessors in den Zellenrechner zur Bewältigung von Echtzeitanforderungen ermöglicht die Durchgängigkeit vom dispositiven Bereich bis zur elementaren Steuerung. Die einzelnen Steuerungskomponenten sind in Bild 54 dargestellt.

Auf dem genannten PC arbeitet unter dem multitaskingfähigen Betriebssystem XENIX das entwickelte Steuerungssystem, das eine automatische Auftragsabwicklung in der Zelle gestattet. Die Auftragsverwaltung überprüft bei Erteilung eines Auftrags, ob die Voraussetzungen für eine Ausführung gegeben sind, fordert entsprechend fehlendes Material und Informationen an, generiert aus dem Rahmenarbeitsplan einen auftragsorientierten Arbeitsplan und leitet betriebsmittel- und materialbezogen Teilaufträge an die Ablaufsteuerung weiter. Dieses zentrale Modul im System koordiniert den Materialfluß zwischen den einzelnen Stationen und erteilt an stations- bzw. materialflußbezogene Steuermodule die nötigen Anweisungen.

In der Stationssteuerung werden Koordinaten und Programmnummern (Steuerdaten) für die Robotersteuerungen (RCM) erzeugt, während die Materialflußsteuerung Schreib- und Leseanweisungen für das Palettenidentifikationssystem (MOBY-M) und Anweisungen für das Transportsystem (SPS) generiert und die Pufferplätze verwaltet. Beide sind an die Gerätesteuerungsebene zur Ausführung ihrer Montage-, Transport- und Identifikationsoperationen angekoppelt. Die Operationen sind aber unabhängig von speziellen Gerätesteuerungen und deren Befehlssyntax. Erst mittels gerätespezifischer Anpaßmodule (pro Gerät eines) werden die Operationen in die entsprechenden Gerätebefehle umgesetzt. Die Anpaßmodule stehen über Pufferbereiche im DCP/MUX-Speicher mit dem entwickelten Kommunikationssystem in Verbindung. Dieses eigenständige Modul auf der DCP/MUX Baugruppe realisiert die Telegrammsequenzen für Gerätebefehle (Interpreter) sowie die Datenübertragung an die Gerätesteuerung (Treiber). So werden Übertragungsprotokolle nach LSV2 - Prozedur und DK3964R eingesetzt.

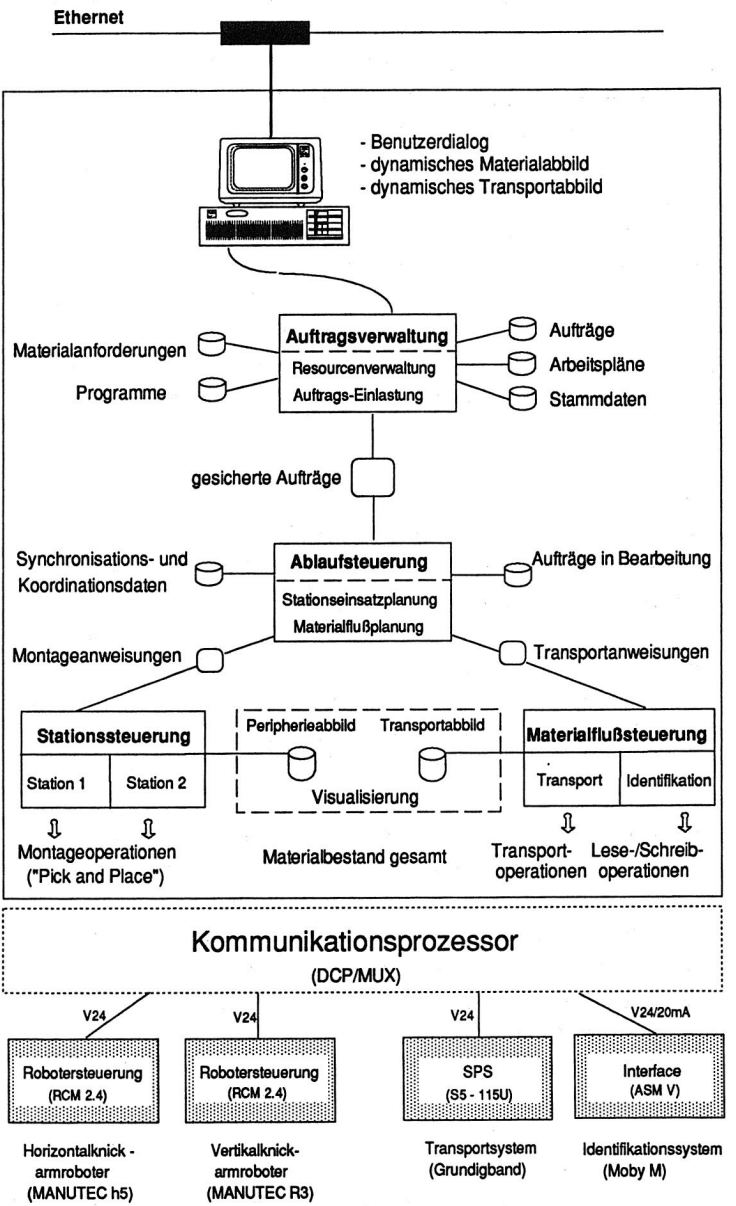


Bild 54: Steuerungskomponenten der Tastaturmontage

Eingesetzte Gerätesteuerungen

Robotersteuerung RCM 2:

Die stationäre Steuerung der Roboter erfolgt mit der Robot Control Machine - RCM 2. Diese Steuerung verfügt im Rahmen der Sensordatenverarbeitung über eine Kopplung /99/, die es ermöglicht, über eine serielle V24-Schnittstelle Daten an die Steuerung zu übermitteln. So können in der Steuerung der Aufruf eines Bearbeitungsprogramms sowie das Verschieben und Verdrehen von Bewegungsabläufen erfolgen.

Speicherprogrammierbare Steuerung (SPS) SIMATIC S5-115U:

Die SPS wird zum Ansprechen und zur Kontrolle des Transportsystems eingesetzt. Für die Kopplung mit externen Rechnern steht der Kommunikationsprozessor CP 525 /99/ zur Verfügung. Er ermöglicht den Zugriff auf die Datenelemente (Eingänge, Ausgänge, Daten- und Programmbausteine) der SPS und ist zuständig für Telegrammübertragung und -interpretation.

Anschaltmodul ASM-V:

Zur Identifikation der Teile befindet sich an den Transportpaletten ein programmierbarer Datenträger, dessen Speicher von einer Lese- und Beschriftungsstation (LBS) induktiv gelesen und beschrieben wird. Diese LBS ist über eine systeminterne, serielle Schnittstelle mit dem Anschaltmodul (ASM-V) /99/ verbunden, das die Daten zwischenspeichert und den Datenverkehr mit dem Rechner abwickelt (über eine 20mA-Linienstromschnittstelle und Umwandlungsstation).

Bei allen eingesetzten Gerätesteuerungen besteht jeder Gerätebefehl aus einer Telegrammsequenz und fristgerechtem Reaktionstelegramm bzw. aus mehreren dieser Sequenzen. Die Initiative für die Datenübertragung kann entweder von einem konkreten Partner ausgehen (z.B. RCM2 oder vom PC beim Ansprechen des ASM-V) oder von beiden Partnern (z.B. SPS - PC), dies erfordert somit weitere Kontrollmechanismen.

DCP/MUX-Kommunikationsbaugruppe:

Sie wurde für den Einsatz in einem AT kompatiblen PC /24/ entwickelt, arbeitet unabhängig vom PC-Prozessor und dient mit ihren acht Ausgängen der Kommunikation mit der angeschlossenen Peripherie. Der PC-Prozessor wird somit entlastet, und freigewordene Rechenkapazität kommt den Anwenderprogrammen zugute. Das Herz der DCP/MUX-Baugruppe bildet ein Intel 80286-Mikroprozessor mit 6MHz Takt, wodurch auf dem AT-Rechner entwickelter Programmcode ebenso ablauffähig ist.

7.2 Entwicklung der Software

7.2.1 Verbindung Funktionshierarchie - Softwaretechnologie

Im vorliegenden Applikationsgebiet 'Flexible Montagezelle' wurden die beschriebenen fertigungstechnischen und softwaretechnologischen Aspekte und Konzepte anhand der konkreten Problemstellung 'Steuerungssystem zur Tastaturmontage' angewandt. In der Analysephase wurden die wesentlichen Elemente bestimmt und ihr Anforderungsprofil festgelegt sowie hierarchische Zusammenhänge und mögliche Ablaufstrukturen beschrieben. Für die Zellenobjekte wurden problemadäquate ADTn erzeugt, so repräsentieren die bestimmten ADTn jeweils Aufgaben innerhalb einer Ebene (Bild 55).

Somit kann sich der Softwareerstellungsprozeß an der Systematik im Anwendungsgebiet orientieren. Die Zuordnungen geben zugleich einen Einblick in die Hierarchie der Anforderungen innerhalb des Softwaresystems. Anforderungen, welche die Stationsprozesse bzw. die Roboterverwaltung erfüllen, werden z.B. benötigt, um Anforderungen der Arbeitsgangabwicklung (Ablaufsteuerung) zu entsprechen. Diese stellen wiederum Subanforderungen der Montageauftragsabwicklung (-verwaltung) dar. Die Analyse wurde deshalb auf Ebene 6 begonnen und dann schrittweise durch die top-down Technik weitergeführt. Die Subanforderungen einer Komponente wurden dabei durch diese Komponente selbst realisiert oder stellten eine (Haupt-) Anforderung einer (in der Abstraktionshierarchie) darunterliegenden Komponente dar.

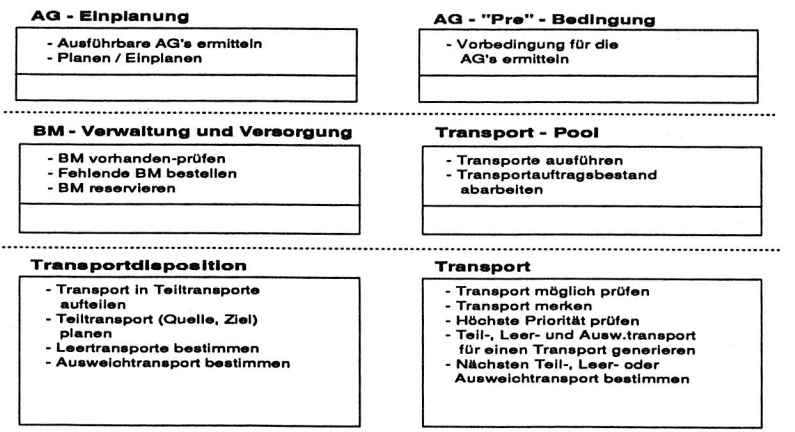


Bild 55: Auszug aus der Hierarchie von ADTn in Ebene 5

Das Vorgehen auf der softwaretechnologischen Seite kann dann sukzessive in top-down Manier fortgesetzt werden bis nur noch Operationen von bereits vorhandenen ADTn oder Basisfunktionen der verwendeten Programmiersprache oder des Betriebssystems verwendet werden. Dadurch kann sichergestellt werden, daß alle Anforderungen, die an die Implementierungsteile der ADTn gestellt werden, auch wirklich realisiert werden.

Die EFFECTS Teile der Operationen neu eingeführter ADTn können dann jeweils bottom-up-mäßig ausgefüllt werden. In integrierten Softwareentwicklungsumgebungen kann direkt aus dem Feinentwurf für eine bestimmte Programmiersprache und einem bestimmten Betriebssystem die Implementierung durch automatische Codegenerierung gewonnen werden. Zur Verwendung einer Bibliothek für die Sammlung von ADTn sei auf Kap. 5.4 und /116/ verwiesen. Als Beispiel kann das Modul 'Transport - Pool' herangezogen werden, das die Ausführung eines Transports auf Arbeitsgangbasis in der Ablaufsteuerung realisiert. Dieses Modul benutzt zur Realisierung die Module 'Transportdisposition' und 'Transport' mit ihren Operationen (Bild 55). Die Materialflußverwaltung erhält nun den generierten logischen Transportbefehl, interpretiert ihn und beauftragt in der ermittelten Reihenfolge einzelne Objekte (Puffer, Übergabeplatz, etc.) zur Ausführung von Transportschritten wie 'Palette einschleusen bzw. ausschleusen'.

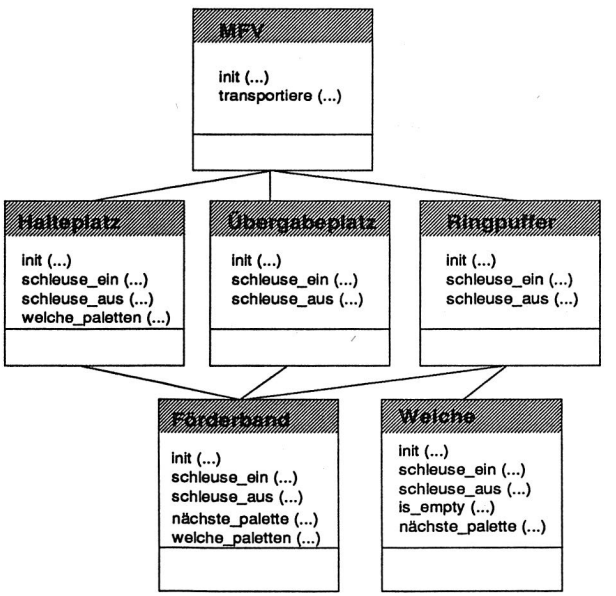


Bild 56: Module der Materialflußverwaltung (MFV)

Modulbeschreibung 'Weiche'

```
MODULE weiche(weichen_i0:I0)

DECLARATION
      weiche AR AS
            inhalt: PALETTE
                  initially:(0,...)
      ENDAR;
      pal     : PALETTE;
      int_ort: IO;        /*interner Ort, von dem die Palette
                            eingeschleust wird*/
      besetzt: boolean;
      empty   : boolean;

TYPES
      PALETTE = record
                        pal_id: integer;
                                :
                  end;
      I0: integer;
CONSTANTS
      nil_palette: PALETTE = (0,...);
           /* entspricht keiner Palette am Ort */
SYN
      x incompatible y : x ∈ SCHLEUSE_EIN ∧ y ∈ SCHLEUSE_AUS;

OPERATIONS

      INIT;
            EFFECTS: inhalt = nil_palette
                              ∧ besetzt = FALSE;

      SCHLEUSE_EIN(pal, int_ort);
            NBL:        besetzt = FALSE;
            EFFECTS:    inhalt = pal
                        ∧ besetzt = TRUE;
            EFFECTS:    /*bewege pal von int_ort
                              nach weichen_io */

      SCHLEUSE_AUS -> pal;
            NBL:        besetzt = TRUE;
            EFFECTS:    pal = inhalt
                        ∧ inhalt = nil_palette
                        ∧ besetzt = FALSE;

      IS_EMPTY -> empty;
            EFFECTS:    if (besetzt = TRUE)
                        then empty = FALSE;
                        else empty = true;

      NAECHSTE_PALETTE -> pal;
            EFFECTS:    pal = inhalt;

END_MODULE /* weiche */
```

Diese Objekte verwenden zur Realisierung dann die von den Modulen Weiche und Förderband zur Verfügung gestellten Methoden (Bild 56). Diese Objekte sind dann direkt in der Lage, die für die Geräteansteuerung (hier SPS) erforderlichen elementaren Transportanweisungen abzusetzen. Beispielhaft wird in Bild 57 der Aufbau des Transportsystems in der Tastaturmontagezelle aus Sicht der Materialflußverwaltung gezeigt.

Für den ADT 'Weiche'von Seite 146 kann als Generierungsparameter eine Kennung, die Nummer des jeweiligen internen Ortes der Weiche (in Bild 57 z.B. 2, 3, 18, 19), angegeben werden. Abstrakt repräsentiert wird die Weiche durch eine Palette (Aufnahme genau einer Palette). Die Typdeklaration wird nicht genauer spezifiziert, da die MFV nur die Palettenidentifikation benötigt, aber nicht den Inhalt (keine unnötige Einschränkung).

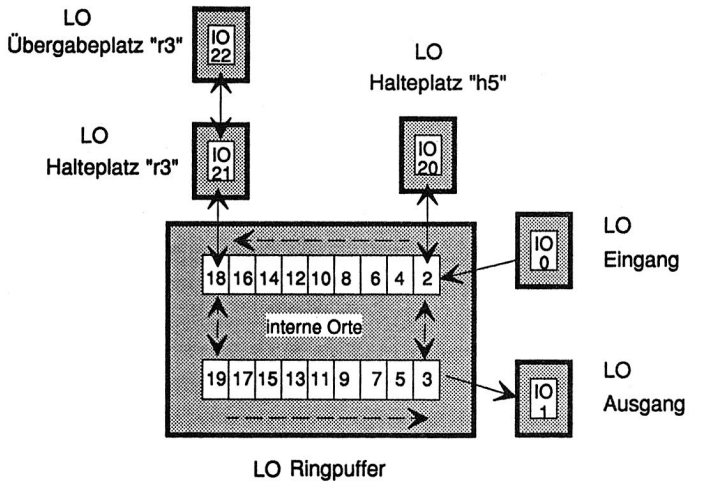


Bild 57: Logische und interne Orte des Transportsystems in der Tastaturmontagezelle

Diese Vorgehensweise entspricht also den im Schichtenmodell aufgestellten Rahmenbedingungen und somit zugleich den softwaretechnologischen Prinzipien der Abstraktion und Hierarchisierung. Beim Moduldesign wurden im Bereich des Applikationsgebietes eine weitere sinnvolle Klassifizierung vorgenommen, eine Unterteilung der Anforderungen in den jeweiligen Ebenen, z.B. gemäß den Komponenten Stationssteuerungen und Transportsystemsteuerung. Das wiederum heißt, Anforderungen, welche die Steuerung der Stationen betreffen und welche die Steuerung des Transportsystems betreffen, wurden nicht gemeinsam in ADTen zusammengefaßt.

Auch die "Objektorientierung" und "Lokalität" wurden insofern berücksichtigt, daß Anforderungen, die unterschiedliche Objekte, wie z.B. Betriebsmittel und Arbeitsgänge, ansprechen, nicht gemeinsam in einem ADT zusammengefaßt wurden. Ein letztes Kriterium für das Moduldesign stellte die Differenzierung der Anforderungen nach den Tätigkeitsbereichen Verwaltung, Planung und Ausführung dar. Auch diese Tätigkeitsgruppen wurden jeweils verschiedenen ADTen zugeordnet.

7.2.2 Integration von Arbeitsplan und Zustandsdaten

Die erforderlichen Montagevorgänge für den Zusammenbau der genannten Modellproduktfamilie wurden analysiert, bestimmten Arbeitsgängen zugeordnet und mit möglichen Ablaufalternativen versehen. Alle diese Angaben wurden in Verbindung mit den jeweiligen Teilklassen in den Strukturgraph integriert.

Ein Rahmenarbeitsplan wurde entworfen und für diese Anwendung implementiert. Bei der Abspeicherung der Informationen wurde darauf geachtet, daß Daten für unterschiedliche Ebenen getrennt selektiert werden können. Der APL untergliedert sich also in $(AG_1(AS_{11}, AS_{12},... AS_{1n})),...(AG_i (AS_{i1},...,AS_{im}))$, wobei jeweils der AS die Operationskennung, den Fügeteil-Ident, den Basisteil-Ident, den Fügeort und einen Technologieparametersatz enthält. Der Übergang vom auftragsneutralen zum auftragsabhängigen erfolgt nach den in Kap. 6.2.4 aufgeführten Algorithmen.

Als Beispiel für die Darstellung spezifischer Arbeitsplaninformationen wird der AG herangezogen, der von der Ablaufsteuerung ausgewertet wird. Notwendige Informationen zur Einplanung und Stationszuordnung betreffen:

- Bereitstellungsort für benötigte Werkstücke (LO-Angabe)
- Betriebsmittel (z.B. Greiferliste und RC-PGM Nummern)
- Koordination von AGs (AG Abhängigkeit)
- Zuordnung AG zu möglichen Stationen (Stationsverweise)

Zur Ansteuerung des Materialflusses wurden alle transportbezogenen Funktionen der Ablaufsteuerung zusammengefaßt. Aus den INPUT- und OUTPUT-Teilelisten (Teilklasse/Teiltyp) und Zielort (LO) werden implizit logische Transportaufträge generiert (dies auch nach Abschluß eines AGs und nach Beendigung aller AGs eines Teilauftrages zur Entsorgung!).

In der Ablaufsteuerung (speziell im implementierten Zellscheduler) werden dann aufgrund möglicher paralleler und gleichzeitiger Ausführung von AGs unterschiedlicher Teilaufträge Arbeitsganglisten mit jeweiliger Zustandseintragung (18 verschiedene AG-Zustände von 'bekannt' bis 'fertig' in einem konzipierten Arbeitsgangdatensatz) geführt /101/. Zur Durchführung von Zustandswechseln werden jeweils aktuelle Daten ausgewertet (z.B Ereignis aus Messages), vgl. hierzu Vorgehensweise aus Bild 27 und 28.

7.2.3 Hilfsmittel zum Entwurf und zur Implementierung

Die Implementierung erfolgte unter dem Betriebssystem XENIX 'System V' in der Programiersprache C. Außer dem CASE System Innovator und den Standardhilfsmitteln, die XENIX bereitstellt, wurde bei der Implementierung noch eine eigens entwickelte Queueverwaltung mit "Monitor"-Programm zur Verfolgung und Protokollierung des Meldungsverkehrs eingesetzt. Diese benutzt die XENIX-Funktionen der Interprozeßkommunikation und stellt den benutzenden Programmen leistungsfähige und einfach zu bedienende Kommunikationswege zur Verfügung. Die Kommunikation zwischen den Prozessen erfolgt über Meldungen mit einheitlichem Format.

Bekannterweise ist im Vorfeld der Anwendung die Simulation (z.B. durch sogenanntes Kurzschließen der Prozesse), das Testen und die Fehlersuche ein gewichtiger Faktor. Im Vergleich zu der Steuerungsentwicklung auf den elementaren Steuergeräten sind die drei genannten Punkte innerhalb einer höheren Programmiersprache leichter zu verwirklichen. Dies ermöglichte somit vorab eine entscheidende Unterstützung zur Überprüfung der Software und der Lösungsmethode im Blick auf die geforderte Anwendung. Unter den generellen Testhilfen (auch zum späteren Einsatz für Initialisierungen) sind Eigenentwicklungen einzuordnen wie z.B.

- Monitortesthilfe zum Verfolgen einzelner Prozeßzustände
- Einspeisen beliebiger Meldungen in den Meldungsverkehr
- Empfangen bestimmter Meldungen aus dem Meldungsverkehr.

Außerdem wurden weitere Testhilfen (Fehler-, Warnungs- und Informationsmeldungen, Fehler- und Datenbasisprotokollierung) eingebaut, die durch Systemparameter gesteuert werden. Über diese Werte können alle Fehler- und Informationsmeldungen selektiv (je nach Testniveau) eingeschaltet werden /101/.

7.3 Darstellung der Zellensteuerung als Prozeßsystem

7.3.1 Informelle Beschreibung

Das Steuerungskonzept für diese Zelle ist durch die Verbindung des Architekturmodells, des Einsatzes bekannter Softwaretools und der Modellierung in Anlehnung an asynchrone Prozeßsysteme entstanden. Im Zellenrechner ergibt sich aus funktionaler, organisatorischer und ressourcenbezogener Sicht (vgl. Kap. 4.3.1) eine Aufteilung in die Module: Auftragsverwaltung (AV), Materialdisposition und -planung (MDP), Zellen-Scheduler (ZS), Stationsplanung (STP), Materialflußplanung (MFP), Stationsverwaltung (STV), Industrieroboter 1 (IR1), Industrieroboter 2 (IR2), Materialflußverwaltung (MFV). Diese Module können direkt als abstrakte synchronisierte Prozeßdatentypen dargestellt und mit einer Verbindungsstruktur versehen werden (Bild 58). Zur Ausführung der Mon-tage werden zusätzlich Objekte (z.B. Betriebsmittel, Arbeitspläne, Material) benötigt, auf denen Operationen (z.B. bestellen, bedarf_an_teilen_ermitteln) ausgeführt werden.

Betrachtet man die Gesamtaufgabe aus der informationsbezogenen Sicht, so kann man zu einer anderen Einteilung gelangen. Objekte, die in den einzelnen Ebenen der Steuerung bearbeitet werden, sind u.a. Aufträge, die in Teilaufträge zerlegt werden. Zur Bearbeitung der Teilaufträge wird eine Folge von Arbeitsgängen, Arbeitsschritten, etc. ausgeführt. Daher könnte man das Prozeßsystem Montagezellensteuerung auch in die PDTen Fertigungsauftrag (FA), Teilauftrag (TA), Arbeitsgang (AG), Arbeitsschritt (AS), Objekt usw. mit ihren Operationen zerlegen. Da jedem Prozeßdatentyp der Aufteilung (AV, MDP, ZS, usw.) ein Datentyp der Unterteilung (FA, TA, AG, ...) zugeordnet werden kann, sind beide Zerlegungen als äquivalent zu bezeichnen. So arbeitet der Prozeßdatentyp AV hauptsächlich mit Fertigungsaufträgen (FA), die MDP entsprechend mit AGs.

Da jedes Modul auf einer anderen Hardware implementiert werden kann, unterstützt diese Zerlegung auch den Einsatz verteilter Systeme für Zellensteuerungen. Die Bedeutung des Kommunikationsbereichs Kom_e betrifft die Verbindung zum übergeordneten Fertigungsleitsystem oder einem Bediener. Sie legen Anforderungen (z.B. FAs) im Kommunikationsbereich Kom_e ab und erhalten von der Auftragsverwaltung hinterlegte Rückmeldungen/Quittungen. Eine verteilte Lösung auf getrennte Prozesse (auch innerhalb einer Ebene) hat im Vergleich zu einer integrierten Lösung folgende Vorteile:

- Alle Funktionen (z.B. zur Bearbeitung von Aufträgen und Materialbeständen) müssen nur einmal realisiert werden und können alle eine gemeinsame Datenbasis benutzen.
- Erweiterungen sind aus konzeptioneller und Realisierungssicht einfacher einzubringen.
- Die Kontrolle des Datenflusses kann bei einer verteilten Lösung dieselben Hilfsmittel benutzen wie für die Kommunikation mit anderen Prozessen.

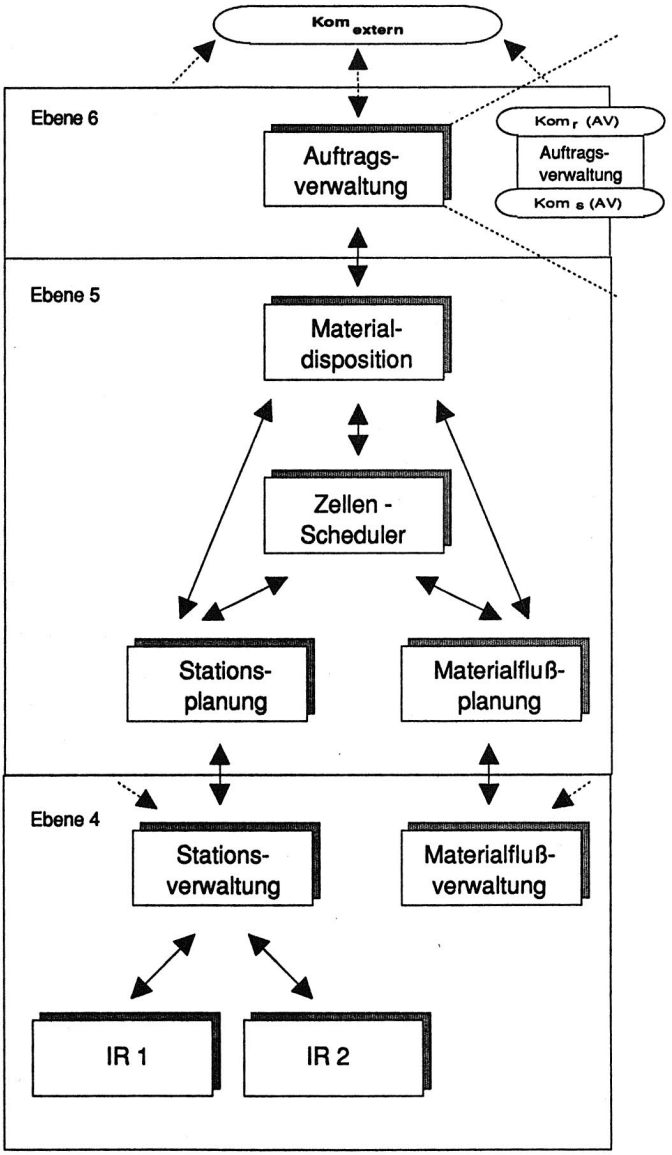


Bild 58: MOST90 als asynchrones Prozeβsystem

Modulbeispiele innerhalb der Ablaufsteuerung

In der MFP sind alle transportbezogenen Funktionen der Ablaufsteuerung zusammengefaßt. Sie bildet die Schnittstelle der Ablaufsteuerung zur Materialflußverwaltung. Sie benutzt deren Angebot an palettenorientierten Transportfunktionen, um den überlagerten Ebenen der Ablaufsteuerung leistungsfähige teileorientierte Transportdienste auf Basis der logischen Orte und Palettenidentifikation zur Verfügung zu stellen. So verbirgt die MFP Inhomogenitäten und Unzulänglichkeiten der zur Verfügung stehenden Transportsysteme. Verwaltung des Transports von Materialträgern (Paletten), Wahl der Transportwege und Generierung konkreter Transporte obliegen der unterlagerten MFV.

Die Stationsplanung bildet die Schnittstelle der Ablaufsteuerung zur Verwaltung und Steuerung der vorhandenen Stationen (hier: Industrieroboter). Diese Planungsart verbirgt die Struktur der Ablaufsteuerung vor der Stationsverwaltung, indem sie dieser als einziger Kommunikationspartner Aufträge von verschiedenen Quellen innerhalb der Ablaufsteuerung vermittelt. So darf z.B. ein Stationsprozeß auf logische Orte nur nach Aufforderung durch die Ablaufsteuerung zugreifen. Handhabungsplätze werden nach ihrer Anzahl von der Ablaufsteuerung verwaltet, die konkrete Zuweisung eines bestimmten Platzes zum Montieren von Teilen obliegt den Stationsprozessen selbst.

7.3.2 Spezifikation des Prozeßdatentyps 'Materialdisposition'

Für eine vollständige Abbildung wurde die Anforderungsdefinition der dispositiven Bereiche der Steuerung aufgestellt und anschließend spezifiert. Anhand der Materialdisposition (MDP) und der Zusammenhänge mit der Auftragsverwaltung (AV) können bestimmte Aspekte der Modellierung mit PDTen verdeutlicht werden. Innerhalb dieses Beispieles soll die Spezifikation nur soweit angegeben werden, wie sie zur Darstellung der wesentlichen Charakteristika des PDT 'MDP' und letztlich zur Darstellung der Montagezellensteuerung als asynchrones Prozeßsystem notwendig ist. Die PDTen 'AV' und 'MDP' wurden zur Spezifikation ausgewählt, da mit ihrer Hilfe die Zusammenarbeit bzw. Kommunikation zwischen zwei Prozeßdatentypen veranschaulicht werden kann und da anhand zweier Module verschiedener Schichten (Abstraktionsniveaus der Zelle) daraus resultierende unterschiedliche Anforderungen aufgezeigt werden können.

In der Spezifikation ist die Realisierung der Kommunikation offengelassen. So ruft z.B. der PDT 'MDP' die Operation 'ta_zustand_akt' des PDTs 'AV' auf, um den Zustand eines Teilauftrags, der beiden Modulen bekannt ist, zu ändern. Die Entscheidung, ob die Zustandsänderung über einen Nachrichtenaustausch im Modul 'AV' erfolgt oder der PDT 'MDP' den Zustand selbst (Shared Memory) aktualisiert, bleibt der Implementierung überlassen. Die 'AV' kann Fertigungsaufträge unabhängig vom Layout der Zelle einplanen. Die 'MDP' muß dagegen Kenntnis über Ausprägung (z.B. LOs) und Stationszustand haben. So beeinflußt z.B. eine defekte bzw. überlastete Station die Zuteilung.

Die Materialdisposition und -planung (MDP)

Die MDP bildet die Verbindung zwischen der Auftragsverwaltung und dem Zellscheduler und umfaßt Funktionen zur Materialdisposition der Stationen sowie Funktionen zur Einplanung neuer Teilaufträge (Bild 59). Nach Erfüllung bestimmter Einplanungsvoraussetzungen (z.B Stationsauswahl) für einen Arbeitsgang wird dieser dem Scheduler zur Ausführung übergeben. Sind alle Arbeitsgänge eines TAs beendet, so meldet die MDP den Teilauftrag als fertig an die AV zurück.

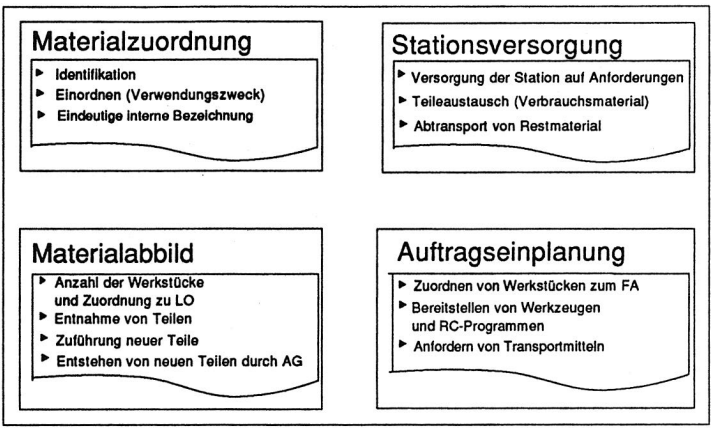


Bild 59: Aufgabenfelder der Materialdisposition

Spezifikation des Prozeßdatentyps MDP

```
MODULE mdp          /* Materialdisposition und -planung */
DECLARATIONS

mdp AR AS
   ta_bestand       : set of teilauftrag;
                    /* Menge der gesicherten Teilaufträge */
   ag_bestand       : set of arbeitsgang; /* Menge der Arbeitsgänge */
   mat_bestand      : set of material; /* Materialbestand */
   station          : integer;
   bestellungen     : set of bestell_anforderung;
   lieferungen      : set of lieferung;
ENDAR;

ta          : teilauftrag;
ag          : arbeitsgang;
teil        : material;
lo1,lo2     : integer;                  /* logische Orte */
bestellung  : bestell_anforderung;
```

```
TYPES

arbeitsgang = record                    /* Arbeitsgang */
    nr              : integer;
    ta_nr           : integer;
    zustand         : (bekannt, in_Disposition, gesichert,
                       ausführbar, eingeplant, begonnen,
                       abgeschlossen, abgebrochen );
    werkstücke      :  set of integer;
              /* Angabe der Identifkation der Werkstücke */
    bm              :  set of integer;
         /* Angabe der Identifikation der Betriebsmittel */
    station         : integer;
    inputteile      : set of material;
    input_bm        : set of betriebsmittel;
  /* Angabe an Werkstücken und Betriebsmittel für AG */
end;

material = record
   pi_nr            : integer; /* eindeutige Identifizierung */
   teilklasse       : integer;
   teiltyp          : integer;
   zustand          : (verfügbar, bestellt, verbraucht,
                       reserviert, in_Bearbeitung )
   aufenthalts_lo : integer
end;

lieferung = record  /* eintreffende Materiallieferung */
   teilklasse : integer;
   erwünscht  : boolean
end;

SYN

x WEAKPRIOR y : x ∈ NEUEN_TA_BEARBEITEN  ∧  y ∈ (BEENDEN_TA
                v AG_AUSWÄHLEN v EINPLANEN v ANF_IRP_BEARB v
                LIEFERUNG_BEARB v MAT_AKT v ABTRANSPORT);

OPERATIONS

/* Einplanung neuer Aufträge */
neuen_ta_bearbeiten(ta);
PRE     :  ta ∈ ta_bestand ∧ ta.zustand = gesichert;
EFFECTS :  mdp.ta_zustand_akt(ta,begonnen)
           ∧ ∀ ag (ag.ta_nr = ta.nr -> ag.zustand = bekannt
                   ∧ ag_bestand = 'ag_bestand + {ag} );
NBL     :  ∀ ag (ag.ta_nr = ta.nr ∧ ag.zustand =abgeschlossen)
           v ∃ ag (ag.ta_nr = ta.nr ∧ ag.zustand=abgebrochen);


                                                /* Teilauftrag beenden */
EFFECTS : if ∃ ag (ag.ta_nr = ta.nr ∧ ag.zustand =abgebrochen)
          then   mdp.ta_zustand_akt(ta,abgebrochen)
          else   mdp.ta_zustand_akt(ta,abgeschlossen);

/* Arbeitsgang bearbeiten, d.h. für die MDP: aus AG-Pool Arbeitsgang
auswählen und notwendige Mittel zuordnen */

ag_auswählen;
CYCLIC;
NBL     :   ∃ ag ∈ ag_bestand (ag.zustand = bekannt);
```

```
/*Arbeitsgang zur Ausführung einplanen mit beliebigen Algorithmen!)*/
EFFECTS :   station_zuteilen(ag,zustandsdaten)
          ∧ werkstücke_zuteilen(ag)
          ∧ rc_progr_zuteilen(ag)
          ∧ werkzeuge_zuteilen(ag)
          ∧ ag.zustand = gesichert;

/* Anforderung von IRP bearbeiten */

anf_irp_bearb;
CYCLIC;
NBL     : (∃ x ∈  bestellungen) ∧(x.von_irp_bearbeitet = true
            ∧ x.von_av_abgesendet = false);
EFFECTS :   mdp.bestell_anf_bearb(x.teilklasse);

/* eintreffende Teile bearbeiten */

lieferung_bearb;
CYCLIC;
NBL     : ∃ x ∈ lieferungen
EFFECTS :  if∃ bestellung ∈  bestellungen
               (bestellung.teilklasse = x.teilklasse
             ∧ bestellung.von_av_abgesendet = true )
           then  lieferung.erwünscht = true
           else  lieferung.erwünscht = false;
EFFECTS : if bestellung.von_irp_angefordert = true
          then teil.aufenthalts_lo = 0
               ∧ mdp.mat_zustand_akt(teil,verfügbar)
               ∧ transport(teilklasse,bestellung.station)
               ∧ lieferungen = 'lieferungen - {lieferung}
               ∧ bestellungen = 'bestellungen - {bestellung}
          else /* Lieferung eines Werkstücks liegt vor */
               teil.pi_nr ermitteln
               ∧ teil.aufenthalts_lo = 0
               ∧ mdp.mat_zustand_akt(teil,verfügbar)
               ∧ bestellungen = 'bestellungen - {bestellung}
               ∧ lieferungen =  'lieferungen - {lieferung};

/* Materialzustand aktualisieren */

mat_akt;
CYCLIC;
/* Transport von lo1 nach lo2 */
NBL     : transportiert(teil,lo1,lo2) = true;
EFFECTS : teil.aufenthalts_lo = lo2;

/* Bearbeitung: altes Teil verarbeitet, neues Teil entstanden*/

EFFECTS : if neu_entstanden(teil)
             then mdp.mat_zustand_akt(teil,verfügbar);

EFFECTS : if bearbeitet(teil)
          then mdp.mat_zustand_akt(teil,verbraucht);

/* Restmaterial abtransportieren */
abtransport;
CYCLIC;
NBL     :∃ teil ∈ mat_bestand ( teil.zustand = unerwünscht);
EFFECTS :  /* Teil aus Zelle entfernen */
```

Unter Verwendung dieser Spezifikation wurde gezeigt, daß das vorgestellte Steuerungskonzept einer FMZ auf ein asynchrones Prozeßsystem abgebildet werden kann. Das asynchrone Prozeßsystem besteht aus den gleichartigen Strukturierungseinheiten 'PDT', die als Modell für die Module des Steuerungskonzeptes verwendet werden können. Die Anforderungen an Initialisierungsmethoden und die Lösungswege zur Erfüllung können aus Kap. 4.6 übernommen und durch Betrachtung der Prozeßdatentypen konkrete Tätigkeiten nachgebildet werden.

7.4 Unterstützung durch relationale Datenbanken

Ein Entwurf eines konzeptionellen Schemas , das einzelnen Ebenen bestimmte Datenbereiche zuordnet (vgl. Kap 6), bildete die Basis für die Festlegung konkreter Schnittstellen innerhalb des Steuerungssystems. Durch Aufstellen von Relationen und durch eine exemplarische Implementierung konnte herausgestellt werden, wie die Generierung von Zellenarbeitsplänen und die Parametrisierung von Arbeitsgängen, Arbeitsschritten, und Aktionen unterstützt wird. Weiterhin wird durch diese Realisierung aufgezeigt, wie Datenströme logisch innerhalb einer Datenbank zusammenlaufen und inwieweit relationale Datenbanksysteme bisher innerhalb von Zellensteuerungen angewendet werden können.

Als Hardware stand eine Apollo-Workstation vom Typ WS-30 zur Verfügung. Beim Betriebssystem handelt es sich um das UNIX-System DOMAIN/IX BSD4.2, das dem UNIX-Standard 4.2 BSD (Berkley-Unix) entspricht. Die Implementierung der Operatorfunktionen wurde in der Programmiersprache C vorgenommen. Die Realisierung des Datenmodells erfolgte auf dem DBMS INGRES, wobei als Sprache SQL eingesetzt wurde /61/.

Ziel der Implementierung war es aufzuzeigen, wie die Abspeicherung der Daten und ihrer Beziehungen untereinander in einer relationalen Datenbank aussehen könnte und in welcher Weise Operatoren als Mittel zum Zugriff auf diese Datenobjekte zur Verfügung gestellt werden können. Da für die anschauliche Realisierung dieser Vorgaben die einfache Bereitstellung von Operatoren innerhalb eines DB-Anwendungsprozesses nicht ausreichend war, wurde auch die Interaktion mit einer Art simulierter Zellensteuerung implementiert. Als Datengrundlage diente die in 7.1 beschriebene Montagezelle.

Die geschaffene Realisierung weist innerhalb eines Programmsystems einen Datenbank-Serverprozeß auf, der der Anwendung Operatoren auf die Datenobjekte der DB zur

Verfügung stellt, die z.B. aufgrund der Montageplanungsergebnisse erlaubt sind. Somit kann der Serverprozeß den Ablauf bei der Abarbeitung von Zellenaufträgen simulieren.

Da alle Prozesse wechselseitig Daten auszutauschen haben, wurde für die Kommunikationsschnittstelle auf die Message-Queueverwaltung der FMZ zurückgegriffen und notwendige Anpassungen durchgeführt. Die Realisierung der Datenbankoperatoren erfolgte in zwei Schritten:

- Isolierte Realisierung der einzelnen DB-Operatorfunktionen und Test auf den realen Anwendungsdaten.
- Integration der Operatoren in den Ablauf der simulierten Steuerung.

Der Umfang sowohl der realisierten Operatoren als auch der simulierten Steuerungsaufgaben wurde auf die wichtigsten Funktionen der Ebenen 6 und 5 beschränkt. Dabei wurden vor allem die Funktionen berücksichtigt, die der Verarbeitung der Aufträge in Teilaufträge, der Auswahl der zu bearbeitenden Arbeitsgänge und der Zuordnung von Stationen, Werkzeug und Werkstücken dienen. Neben Protokollfunktionen wurde außerdem auf Ebene 4 die Generierung der AS-Parameter in die Realisierung mitaufgenommen /61/. Den Überblick über die Implementierung zeigt das Ablaufdiagramm in Bild 60 mit den Zugriffen zur Datenbank und den Dialogaktivitäten des Anwenders.

Der in der Realität auf ein komplexes Prozeßsystem aufgeteilte Steuerungsablauf wurde sehr stark in einen linearen Ablauf von Auftragseingang bis Erstellung einer parametrisierten AS-Liste komprimiert. Für konstruierte Aufträge stehen die für den gesamten Vorgang benötigten Daten komplett zur Verfügung. Ausgangspunkt bildet dabei jeweils ein definierter Zellenzustand.

Da in den Steuerungsprozeß Dynamik einbezogen werden sollte, wurden die Auswahlvorgänge, die im Realfall die Steuerung selbst übernimmt, dem Benutzer übertragen. Hierbei leistet die von INGRES zur Verfügung gestellte maskenorientierte Schnittstelle gute Dienste. Für jeden Auswahlvorgang wurde eine Bildschirmmaske definiert, die die aus der DB gelesenen Daten aufnimmt und dem Benutzer die cursorgesteuerte Wahl eines der Datenelemente oder einfach den Überblick über die gelesenen Daten gestattet.

Die vorgegebenen Rahmenbedingungen für die Implementierung des Datenmodells ermöglichen hier nur eine Bewertung der 'dispositiven' Teilaspekte der Datenbankanwendung in flexiblen Montagezellen. Hierbei wurden auch nicht Realzeitaspekte oder ähnliche Kriterien für den Einsatz in realen Zellensteuerungen untersucht.

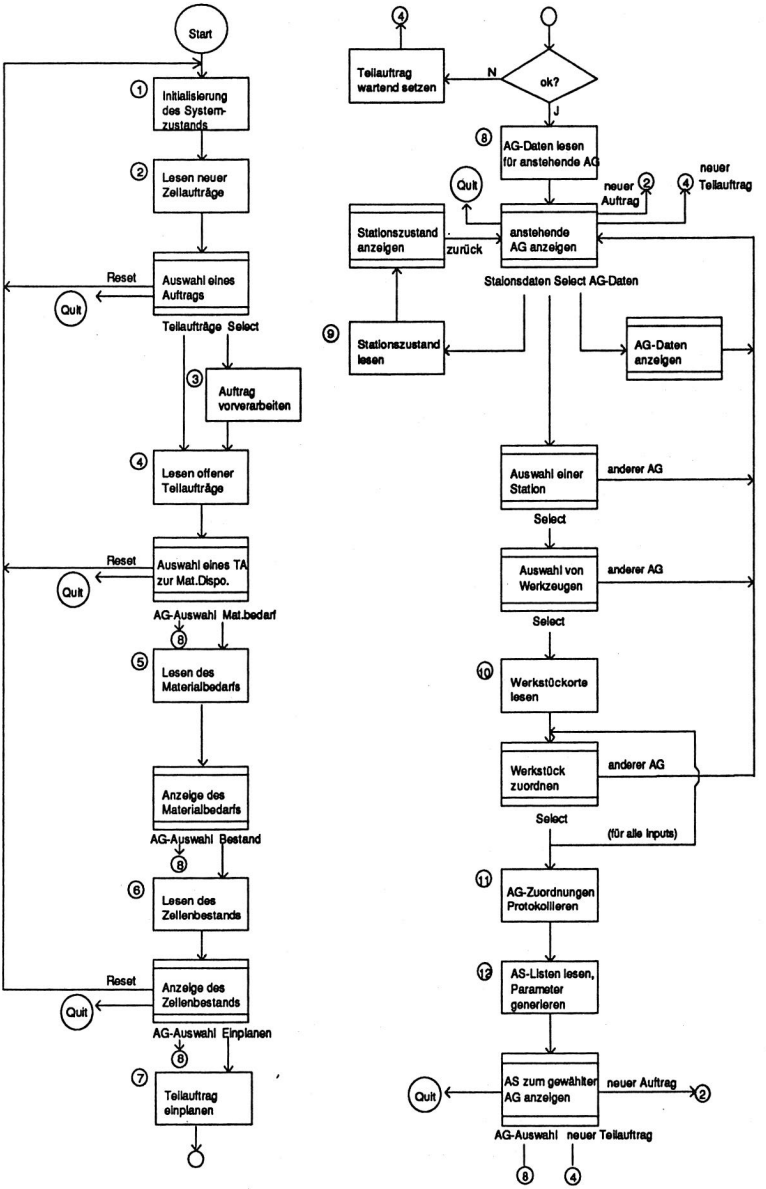


Bild 60: Implementierungsschema einer DB-Anwendung in bezug auf das Ebenenmodell

Vielmehr wurde aufgezeigt, wie die Datenströme aus den verschiedenen Bereichen - Stammdaten, Auftragsdaten, Rahmenarbeitspläne und Zustandsdaten - in der Datenbank zusammengefaßt werden können und wie dabei die Umsetzung eines konzeptionellen Modells in eine Menge von Datenobjekten auszusehen hat. Das relationale Datenbanksystem INGRES hat sich unter diesen Rahmenbedingungen als geeignet erwiesen, das erarbeitete konzeptionelle Datenschema in eine Menge von Relationen umzusetzen. Zur Optimierung des Zugriffs (Zugriffszeiten, Aspekt der konkurrierenden Zugriffe) bietet DBMS verschiedene Zugriffs- und Speicherungsstrukturen (Hashtabellen, Indextabellen, B-Bäume) an.

Die vorgenommene Implementierung hat gezeigt, daß das Transaktionskonzept, wie es in INGRES verwirklicht ist, gut geeignet ist, die Sicherheit der Daten zu gewährleisten. In Verbindung mit der Definition von Integritätsbedingungen stellt das Transaktionskonzept ein wichtiges Hilfsmittel für die Konsistenzerhaltung der Daten dar. Integriert man zusätzlich bekannte Möglichkeiten der Datenvorverteilung zur Verbesserung der Reaktionszeiten, so können horizontale oder vertikale Fragmente von Relationen, aus Gründen der Datensicherheit jeweils repliziert, in die Zugriffsbereiche der Prozesse transferiert werden.

Das 'Snapshot - Prinzip' erscheint im Vergleich zum 'Kopie - Prinzip' für eine Montagezelle wegen ständig stattfindender Datenänderungen geeigneter. Es findet hierbei eine regelmäßige Auffrischung der 'read-only' Datenbestände von der Datenbankseite statt, und von Zeit zu Zeit (z.B. Beendigung eines Teilauftrages) werden dann die Änderungen in der Zelle dem Datenbanksystem direkt mitgeteilt. Als Beispiel für ein Konstruktionsdiagramm und Begriffsschema sei der Objekttyp Werkstückbestand herangezogen.

```
WERKSTÜCKBESTAND (Werkstück Nr., Ausprägungs Nr., Status,
                  Auftrags Nr., Werkstückträger Nr., Ort auf
                  Werkstückträger, Basis/Fügeteil, Lagerort)
```

Als Attributstypen gelten z.B für Status reserviert, frei, bestellt, defekt und für Lagerort die Angaben über LO und LLO. Hieraus lassen sich dann Integritätsbedingungen für diesen Objekttyp (vgl. Kap. 5) ableiten und auch Beziehungsrestriktionen zu anderen Objekttypen (z.B. MATERIALSTAMM) finden. Der jeweilige lokale Datenbestand für die einzelnen Ebenen (horizontale und/oder vertikal fragmentierte Relationen) wird jeweils mittels SQL beschrieben. Die Spalten einer Relation werden mittels der SELECT Anweisung über ihren Namen angesprochen und herausprojeziert; bestimmte Zeilen werden dagegen über ihren Inhalt, nämlich mittels den in der WHERE Anweisung angegebenen Prädikaten qualifiziert.

Beispiel: lokaler Datenbestand für Werkstückreservierung

```
SELECT Werkstück Nr., Ausprägungs Nr., Status,
       Auftrags Nr., Lagerort
FROM   Werkstückbestand
WHERE  Status = 'frei' v Status = 'bestellt'
```

Für die Komplexität der Anwendungen im Bereich der flexiblen Montage unterstützt der Einsatz relationaler Modelle den neuen Lösungsansatz für die Entwicklung geeigneter Zellensteuerungen. Die angestrebte Transparenz des Steuerungsvorgangs kann durch eine geeignete Konzeption der zugehörigen Datenstrukturen und Datenhaltung verstärkt werden. Das Datenmodell wurde speziell auf die maximale Flexibilität des Systems ausgerichtet, d.h. es wurde darauf geachtet, daß sich keine Beschränkungen in der flexiblen Abarbeitung der Aufträge ergeben.

Analog zur hierarchischen Gliederung der Steuerungsfunktionen stellt das Datenmodell jeweils die auf den verschiedenen Ebenen benötigten Elemente des Rahmenarbeitsplans zur Verfügung. Soweit als möglich erfolgte eine Trennung bei der Bearbeitung der Daten zu den verschiedenen Ebenen, um die Bildung von Schnittstellen zu unterstützen. Die Daten wurden so strukturiert, daß onlinemäßig die Parametrisierung der Arbeitsplanelemente der jeweiligen Ebenen möglich ist. Die Einbeziehung von Zustandsdaten in das Datenschema unterstützt die flexible Abarbeitung der Aufträge.

7.5 Anwendung im Hinblick auf methodisches Vorgehen

Die methodische Erstellung einer Steuerungssoftware ist im Bereich der Montageautomatisierung von besonderer Bedeutung, nicht zuletzt aufgrund der Integration ständig neuer Anforderungen. Beispielsweise kann die Einführung einer neuen Produktfamilie verschiedene Änderungen bestehender Geräte- und Steuerungseinheiten erfordern. Aus diesem Grunde sollte sich die Entwicklung einer Steuerungssoftware von vornherein am entwickelten logischen Modell orientieren, das auch bei neuen Anforderungen Gültigkeit besitzt.

Ferner sollte die klare Trennung verschiedener Entwicklungsphasen erfolgen (vgl. 4.3.1). Die Ergebnisse dieser Phasen sind sowohl für eine spezielle Entwicklung als auch für allgemeine Aspekte von besonderem Nutzen. So können schließlich unter Beachtung von vorgegebenen Gliederungsgesichtspunkten grundlegende Module, die einmal erstellt wurden, beibehalten oder mit vertretbaren Aufwand angepaßt werden.

Das logische Modell der Abstraktionshierarchie stellte die Grundlage für die Entwicklung von 'MOST90' sowie für die Erweiterung und Änderung dieser Steuerungssoftware dar. Das Prozeβsystem 'MOST90' wurde auf einen Montagezellenrechner übertragen und angepaβt, der eine Komponente zur Präsentation von CIM-Aspekten innerhalb eines flexiblen Produktionssystems bildete. Hierbei steht das Zusammenwirken einer Flexiblen Montagezelle, einer Flexiblen Drehzelle und einer Spritzguβmaschine einschlieβlich der jeweiligen Steuerungseinheiten im Vordergrund (Bild 61).

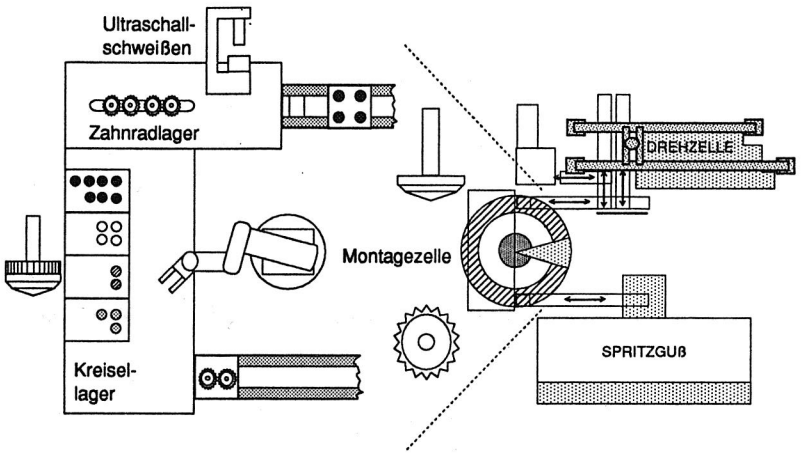


Bild 61: Layout des Flexiblen Produktionssystems

Als Beispielprodukte werden Kreisel - bestehend aus Kegeln und Zahnrädern in unterschiedlichen Varianten - montiert. In der Flexiblen Drehzelle werden Kegel entsprechend vorgegebener Fertigungsaufträge in bestimmter Anzahl und von bestimmter Sorte gedreht. Fertige Drehteile werden der Montagezelle auftragsgesteuert über ein Palettentransportband zugeführt.

Unabhängig des Auftragsbestandes der Dreh- und Montagezelle fertigt die Spritzguβmaschine Zahnräder für unterschiedliche Abnehmer. In der Montagezelle werden diese neben den gelieferten Kegeln zur Montage der Enderzeugnisse 'Kreisel' benötigt. Die Bestellungen erfolgen verbrauchsgesteuert (Bestandsführung im Zellenrechner) . Dieser Nachschub wird montagegerecht per Stangenmagazine (jeweils zwei auf einer Palette) über ein zweites Transportband zum Übergabeplatz der Montagezelle transportiert und steht dann dort zur Entladung bereit.

Die Montagesteuerung hat im Rahmen der Flexiblen Montagezelle die Aufgabe, das Entladen des Zahnradnachschubs bzw. die Montage der gelieferten Kegel planmäßig einzuleiten. Die jeweils vom Roboterprogramm benötigten Koordinatenwerte werden entsprechend des Peripheriezustandes ermittelt. Das Roboterprogramm mit den aktuellen Parameterwerten sorgt dann dafür, daß der Roboter den Zahnradnachschub auf den dafür vorgesehenen Lagerplatz entlädt bzw. die gelieferten Kegel der Reihe nach auf dem Montageplatz justiert, mit Zahnrädern bestückt und die fertigen Kreisel je nach Sorte auf die vorgesehenen Ablageplätze transferiert. Durch den Roboter geleerte Paletten werden zur Drehzelle bzw. zur Spritzgußmaschine zurücktransportiert, um dort entsprechend dem Bedarf und den Fertigungsmöglichkeiten erneut beladen zu werden.

Zunächst wurden anhand des hierarchisches Architekturmodells die konkreten Aufgabenbereiche der verschiedenen Ebenen beschrieben. Anschließend wurde die erweiterte Softwareerstellungsmethode aus Kap. 4.3.1 praxisnah eingesetzt. Anhand von Beispielen werden nun Vorgehensweise und Ergebnisse der einzelnen Phasen für diese Steuerungssoftware erläutert.

7.5.1 Durchgängigkeit von Spezifikation und Implementierung

Zur Strukturierung der teilweise ähnlichen Anforderungen im Vergleich zu MOST90 werden die einzelnen Aufgaben zerlegt und bestimmten Ebenen zugeordnet. Bei diesen hierarchisch organisierten Bearbeitungsebenen werden Teilaufgaben einer Ebene nur von Teilaufgaben höher gelegener Ebenen oder von der gleichen Ebene angestoßen. Von unterlagerten Ebenen werden nur Bestätigungen für ausgeführte Teilaufgaben empfangen oder Signale, daß neue Teilaufgaben abgesendet werden können.

Als weiteres Mittel zur Erhöhung der Überschaubarkeit und einer späteren Möglichkeit, das System leicht zu ergänzen bzw. zu ändern, werden in der Phase des Grobentwurfs Formalismen wie in der Phase des Feinentwurfs benutzt; Operationen, Prädikate und Datenvorräte werden aber nur informell zum Ausdruck gebracht, d.h. Konkretisierungen werden offengelassen. Dadurch wird ein Zwischenergebnis der Spezifikation erreicht, das sowohl für den Ersteller der Anforderungsdefinition als auch für den Ersteller der Feinspezifikation eine Diskussionsgrundlage bildet.

Der Ersteller der Anforderungsdefinition kann anhand der Grobspezifikation die Inhalte, logischen Zusammenhänge und Ablauffolgen leichter überblicken, prüfen und gegebenenfalls ändern bzw. ergänzen. Eine Änderung an dieser Stelle ist - bedingt durch die

höhere Abstraktionsstufe - auch für den Ersteller der Feinspezifikation mit weniger Aufwand hinzunehmen als an späterer Stelle, wenn die Feinspezifikation oder die Implementierung vorliegt.

Anforderungsdefinition

Am Beispiel Auftragsverwaltung und Ablaufsteuerung werden Anforderungsdefinitionen erstellt. Auf Grundlage des Schichtenmodells und der Komponentenermittlung aus 4.3.3 wird die Frage "Was ist zu tun?" (Bild 62) in die Fragen "Was kann sich ereignen?" und "Was ist daraufhin zu tun?" zerlegt. Durch die erste Frage wird geklärt, was für Informationen von welchen Komponenten auf die verschiedenen Ebenen zukommen. Durch die Zerlegung können dann notwendige Konsequenzen, die aus den Informationen zu ziehen sind, erkannt werden.

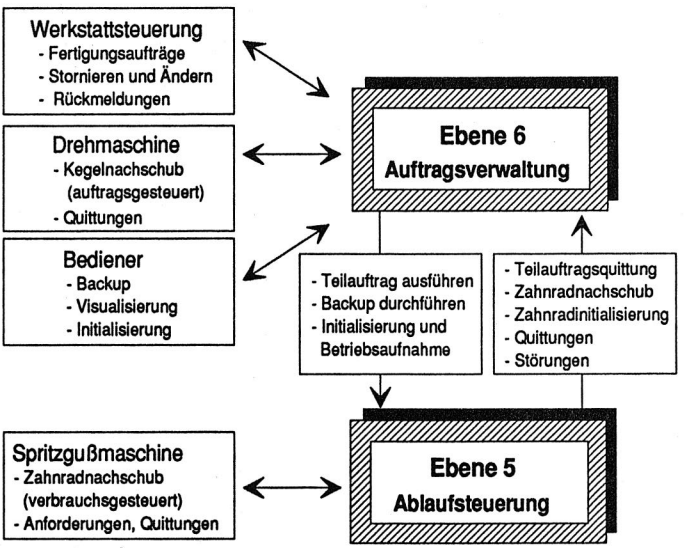


Bild 62: Anforderungen an Auftragsverwaltung/Ablaufsteuerung

Grobspezifikation

Der PDT 'auftragsverwaltung' wird in der Grobspezifikation aufgrund der informellen Anteile leicht verständlich. An Auszügen wird aufgezeigt, wie dieses Ziel erreicht wurde.

MODULE auftragsverwaltung;

```
DECLARATIONS

auftragsverwaltung AR AS
   Arbeitszustand             : set of zustand;
   Fertigungsauftragsbestand  : set of fertigungsauftrag;
   Teilauftragsbestand        : set of teilauftrag;
   Arbeitsplanbestand         : set of arbeitsplan;
   Kegelbestand               : set of kegel;
   Zahnradbestand             : set of zahnrad;
                    ENDAR;

INIT:     Arbeitszustand              = {Bearbeitungszustand}
    AND   Teilauftragsbestand         = {}
    AND   Arbeitsplanbestand          = {Rahmenarbeitsplan}
     ...
```

Bei der Grobspezifikation wird auf Datentypen verwiesen, die nicht näher erläutert werden. Auf einen TYPES-Teil wird verzichtet, d.h die Typen bleiben abstrakt. Im INIT-Teil sind keine Initialisierungen für Auftragsbestand, Kegelbestand und Zahnradbestand angegeben, d.h. diese Bestände können je nach Anlagenspezifika später angegeben werden.

```
SYN
x compatible y  :      x ∈  KEGELNACHSCHUB_EINTRAGEN
                       y ∈ (FERTIGUNGSAUFTRAG_AUSFÜHREN,
                            NEUER_FERTIGUNGSAUFTRAG,
                            FERTIGUNGSAUFTRAG_LÖSCHEN,
                            ZAHNRADBESTAND_ERHÖHEN,
                            BACKUP)
  ...

x  prior  y     :      x ∈ (BACKUP_BEARBEITEN,
                            NEUER_FERTIGUNGSAUFTRAG,
                            FERTIGUNGSAUFTRAG_LÖSCHEN)
                       y ∈  FERTIGUNGSAUFTRGAG_AUSFÜHREN
```

Alle Operationen, die kompatibel sind, dürfen gleichzeitig aktiv sein. Alle nicht aufgeführten Kombinationen sind nicht kompatibel. Die Prioritätenregelung sorgt dafür, daß 'fertigungsauftrag_ausführen()' nicht aktiviert wird, wenn eine der als prior angeführten Operationen aktiviert werden kann.

```
OPERATIONS

fertigungsauftrag_ausführen;
CYCLIC;
NBL:        Bearbeitung ∈ Arbeitszustand
        AND Ausführung ∉ Arbeitszustand
        AND Kegelbestand nicht leer
        AND Zahnradbestand nicht kleiner als Kegelbestand
```

```
EFFECTS:    Ausführung in Arbeitszustand eintragen;
EFFECTS:    IF ( Existiert kein zum Kegelbestand existieren-
                 der, passender Fertigungsauftrag)
              THEN
                 Kegelbestand zum Erstellen eines
                 Fertigungsauftrages verwenden
                AND Interne Parameter für FA bestimmen
                AND FA in den Auftragsbestand einordnen;
EFFECTS:    Sicheren Teilauftrag zu einem FA des Auftrags-
            bestands mit passendem Kegelbestand erstellen
            und in den Teilauftragsbestand eintragen;
EFFECTS:    Arbeitsplan zum Sicheren Teilauftrag
            aus Rahmenarbeitsplan erstellen und in den
            Arbeitsplanbestand eintragen;
EFFECTS:    Ablaufsteuerung soll Sicheren Teilauftrag nach
            Arbeitsplan ausführen;

fertigungsauftrag_löschen(Löschparameter)
                          -> Fertigungsauftrag;
PRE:        Existiert ein FA im FA-Bestand, dessen Parameter
            den Löschparametern entsprechen
NBL:        Bearbeitung ∉ Arbeitszustand
EFFECTS:    Streiche diesen FA aus dem FA-Bestand;
```

Die Menge und Art der 'Löschparameter' ist noch nicht festgelegt. Damit bleibt noch offen, nach welchen Kriterien ein Fertigungsauftrag gelöscht werden kann. Bei der Grobspezifikation der Ablaufsteuerung soll hier innerhalb des PDT 'ablaufsteuerung' die Operation 'arbeitsgang_ausführen' erwähnt werden.

MODULE ablaufsteuerung;

```
DECLARATIONS ...

OPERATIONS

arbeitsgang_ausführen;
CYCLIC;
NBL:        Bearbeitung ∈  Arbeitszustand
        AND Ausführung ∉ Arbeitszustand
        AND Existiert ein wartender Fügearbeitsgang
            oder Entladearbeitsgang im Arbeitsgangbestand;
EFFECTS:    Ausführung in Arbeitszustand eintragen;
EFFECTS:    IF (Fügearbeitsgang wartet)
              THEN Fügearbeitsgang aktiv
                                   /* Entladearbeitsgang wartet */
              ELSE  Entladearbeitsgang aktiv;
EFFECTS:    Roboterverwaltung soll aktiven Arbeitsgang des
            Arbeitsgangbestandes ausführen;
```

Feinspezifikation

Von der Feinspezifikation für die Auftragsverwaltung und Ablaufsteuerung wird hier das Beispiel 'sicheren_ta_erstellen' angegeben. Datentypen werden festgelegt, Operationen entsprechend formalisiert und durch weitere Operationen ergänzt. Das Abstraktionsni-

veau der Feinspezifikationen ist aufgrund der konkretisierten Datentypen und Operationen niedriger als das der Grobspezifikationen und kann sich schon mit direkten Implementierungsdetails beschäftigen.

MODULE auftr;

```
DECLARATIONS ....

TYPES
zellenauftrag = record
    nummer                : integer;
    priorität             : integer;
    kreiseltyp            : integer;
    kreiselzahl           : integer;
    ta_fertig_zahl        : integer;
    kreisel_je_ta_fertig  : set of integer;
    restkreiselzahl       : integer;
    ta_zahl               : integer;
    kreisel_je_ta         : set of integer
end;  ...
```

Festlegungen der Datentypen in der Feinspezifikation sind 'richtungsweisend' für die anschließende Implementierung.

```
OPERATIONS

sicheren_ta_erstellen() -> ta;
NBL:       ( ∃ fa ∈ fa_bestand :
           ( fa.restkreisel.zahl ≥ kegelbestand.zahl
             AND fa.kreisel.typ = kegelbestand.typ  ))
EFFECTS:       ta = new(teilauftrag)
            AND ta.ausgeführt = false
            AND ta.fa_nr = fa.nummer
            AND ta.nummer = fa.ta_fertig_zahl + 1
            AND ta.kreiselklasse = fa.kreiseltyp
            AND ta.kreiselzahl = kegelbestand.zahl
            AND ta.zahnradzahl = kegelbestand.zahl
            AND ta.kegelklasse = kegelbestand.typ
            AND ta.kegelzahl = kegelbestand.zahl;
EFFECTS:       ta_bestand = 'ta_bestand + {ta};
```

Implementierung

Bei der Implementierungsphase müssen sämtliche Hard- und Softwarevoraussetzungen berücksichtigt werden. Die Grobspezifikation und die Feinspezifikation wurden in dieser Arbeit völlig unabhängig von diesen Voraussetzungen erstellt. Damit liefern diese Spezifikationen wertvolle Ergebnisse für unterschiedlichste Anwendungssysteme. Die folgende Implementierung wurde in der Programmiersprache C auf einem Einprozessorsystem unter dem multitaskingfähigen Betriebssystem XENIX durchgeführt. Der Meldungsverkehr zwischen den verschiedenen Prozessen wird durch ein softwaremäßig installiertes Messagequeue-Konzept (vgl. MOST90) geregelt.

Für die Komponenten Werkstattsteuerung, Auftragsverwaltung, Ablaufsteuerung, Kegel- und Zahnradeingabe wurde jeweils ein Prozeß erstellt. Der beschriebene abstrakte synchronisierte Prozeßdatentyp wird insofern eingeschränkt benutzt, als daß seine Operationen einem Prozeß zugeordnet werden und diese damit sequentiell abgearbeitet werden. Nach Meldungsempfang werden nach der Reihe sämtliche Operationen angestoßen, die für die Verarbeitung der Meldung von Bedeutung sind. Erst nach dieser Verarbeitung wird eine neue Meldung entgegengenommen.

Es wird eine Abarbeitungsfolge festgelegt, die in der Spezifikation zum Teil noch aufgrund von Synchronisationen vorgenommen wurde. In der Spezifikation konnten z. B. beim Prozeßdatentyp 'auftragsverwaltung' die Prozeduren 'kegelnachschub_eintragen()' und 'zahnradbestand_erhöhen()' unter bestimmten Bedingungen gleichzeitig aktiviert werden.

Umsetzen der Feinspezifikation an einem Beispiel

Die Implementierung der Auftragsverwaltung orientiert sich wie bei allen Prozessen an der erläuterten Feinspezifikation. Die Operationen des PDT wurden als zu konkretisierende Prozeduren übernommen. Sämtliche Datentypen und Operationen wurden endgültig im Rahmen der Implementierungsmöglichkeiten festgelegt. Nichtblockierungsbedingungen fanden als Kontrollstrukturen Verwendung. Die Programmstruktur wurde zyklisch und entsprechend der Operationen der Grob- und Feinspezifikation aufgebaut. Am Beispiel der Operation 'fertigungsauftrag_löschen (nummer)' wird die Abgrenzung der Implementierung von der Feinspezifikation aufgezeigt.

Die Operation 'kegelnachschub_eintragen()', die schon aus der Grobspezifikation bekannt ist, findet zum Beispiel als konkretisierte Prozedur Verwendung. Die Prozedur 'fertigungsauftrag_ausführen()' war bei der Grobspezifikation als Prozeß bekannt. Bei der Feinspezifikation werden innerhalb dieses Prozesses die Operationen 'neuer_fertigungsauftrag()', 'sicheren_ta_erstellen()' und 'apl_erstellen()' aufgerufen und spezifiziert.

Bei der Implementierung mußten aufgrund des Meldeverkehrs Meldungen gelesen und gesendet werden, so daß erstmalig die Prozeduren 'kegel_meldung _lesen()' und 'apl_senden()' auftauchen. Die erste Prozedur liest die für die Ausführung der Prozedur 'kegelnachschub _eintragen()' benötigten Informationen aus der speziellen Meldung ein. Die zweite Prozedur trägt die Informationen, die von der Ablaufsteuerung zur Ausführung der Prozedur 'ta_ausführen()' benötigt werden, in eine neue Meldung ein und

sendet diese einschließlich aller Attribute (Sender, Empfänger, Klasse) ab. Der Unterschied zwischen Implementierung und Feinspezifikation liegt nicht in der Menge der verfügbaren Prozeduren bzw. Operationen, sondern vielmehr in den Prozeduren selbst.

Zum Vergleich mit der Implementierung wird beispielsweise die in der Feinspezifikation angegebene Prozedur 'fa_einordnen()' als Teiloperation von 'neuer_fertigungsauftrag()' herangezogen. Hierbei ist der Datentyp 'fertigungsauftrag' auf Implementierungsebene in C als '*typedef struct fa*' festzulegen (auch bei Tool - Unterstützung). Für Feinspezifikation und Implementierung wird jetzt die genannte Prozedur mit Bezug auf die entsprechenden Datentypen beschrieben. Diese Teile beschränken sich auf das Einordnen von Fertigungsaufträgen mit hoher Priorität.

An diesem einfachen Beispiel wird der Unterschied zwischen der noch abstrakt gehaltenen Feinspezifikation und der konkreten, mit bestimmten Techniken arbeitenden Implementierung deutlich. Während bei der Feinspezifikation mit der einfachen Operation '+' in Verbindung mit dem Datentyp 'set of fertigungsauftrag' das Ziel der Aufgabe erreicht wird, muß bei der Implementierung der Aufbau einer Fertigungsauftragsliste mit berücksichtigt werden.

Feinspezifikation:

```
pr_hoch     : identifier;
fa          : fertigungsauftrag;
fa_bestand  : set of fertigungsauftrag;

fa_einordnen(fa);
EFFECTS:     IF (fa.priorität = pr_hoch)
               THEN
                 fa_bestand = {fa} + 'fa_bestand';
               ELSE ...
```

Implementierung:

```
fa_einordnen(fa,fa_erst,fa_letzt,priorität)
FA *fa,fa_erst,fa_letzt;
int priorität;
{
if (priorität = 1)
  {
  fa->nachfolger = fa_erst;
  if(fa_erst != NULL)
     fa_erst->vorgaenger = fa;
  fa->vorgaenger = NULL;
  if(fa_letzt == NULL)
     fa_letzt = fa;
  fa_erst = fa;
  }
else ...
```

7.5.2 Wiederverwendungsaspekte

Das Ziel der Wiederverwendung von Software ist die Umsetzung des in der Elektronikindustrie bekannten Prinzips des "Shopping and Making". Erste Schritte auf diesem Weg sind Erkenntnisse, daß wiederverwendbare Bausteine nicht nur auf den Programmcode /26/ bezogen werden, sondern auch auf jede Form eines Entwurfs- und Spezifikationsproduktes. Je nach Sichtweise und Anforderungen des Benutzers ist prinzipiell Material auf jeder beliebigen Abstraktionsebene wieder einsetzbar. Die Chancen der Wiederverwendung bestehender Erkenntnisse und vorhandenen Materials in einer anderen Umgebung sind umso besser, je abstrakter die jeweiligen Komponenten sind.

Geeignete Mechanismen zur Unterstützung der Wiederverwendbarkeit (Abstraktion, Information-Hiding, Modularisierung) finden sowohl im Architekturmodell als auch im vorgestellten Prinzip der ADT Verwendung. Erst mit Hilfe beider Aspekte lassen sich Komponenten eines Problembereiches in Softwarekomponenten (Widerspiegelung des Aufbaus und der Struktur des Applikationsgebietes) übertragen, was für eine Akzeptanz dieser Module von größter Bedeutung ist. Dem jeweiligen Anwendungsgebiet angepaßte Terminologie und Datenstrukturen bilden dann die Grundlage für den Aufbau einer Sammlung geeigneter ADTs (z.B. Aufbau einer Bibliothek).

Neben der Problematik der Eindeutigkeit von Begriffen wie "Puffer, Warteschlange, Magazin" spielen aber auch eine Reihe psychologischer Faktoren eine große Rolle (z.B. das 'Selbermachen-Phänomen' oder die 'Black-Box Angst'). Der Unterschied im Projektverlauf liegt also in der gezielten Verwendung bereits vorhandenen Materials in bezug auf Anforderungen des Projektes und in einer evt. Aufbereitung neuer Erkenntnisse. Eine systematische und methodische Vorgehensweise (z.B. Einsatz bekannter Zerlegungsmechanismen oder bereits getesteter -d.h. weniger fehleranfälliger- Codemodule) beeinflußt sicherlich positiv sowohl die Produktivität als auch die Qualität der erstellten Software.

Innerhalb des aus der Anwendung bekannten PDTs 'Industrieroboterverwaltung' stellt die Materialverwaltung einen Hauptaspekt dar. Magazine in unterschiedlichsten Ausprägungen sind stets als Strukturen in der Peripherie vorzufinden. Um nicht nur die Hardwarekomponenten, sondern auch zugehörige Steuerungskomponenten im Zellenrechner in unterschiedlichen Anwendungen zu nutzen, wird am Beispiel 'Magazin' eine mögliche Vorgehensweise aufgezeigt und mit den Anforderungen aus den vorher aufgeführten Zellenrechnersystemen verknüpft.

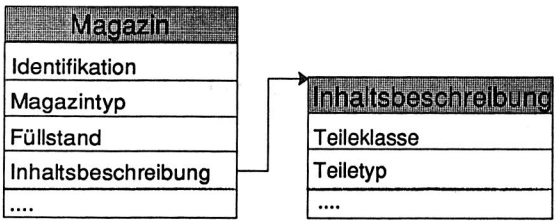


Bild 63: Datenstruktur 'Magazin'

Um zur Magazinverwaltung gehörende Funktionen wiederverwendbar zu machen, müssen auch Datenstrukturen zur Beschreibung der Magazine erweiterbar und änderbar sein. Eine Lösungsmöglichkeit im Rahmen der Prototypenerstellung, die diesen Anforderungen genügt und zusätzlich auch eine Ergänzung durch andere Algorithmen zuläßt, zeigt Bild 63.
Die Datenstruktur enthält keine direkten Angaben über die Geometrie des Magazins. Erst anhand des 'Magazintyps' (Bild 64) erhält man Werte über physikalische Gegebenheiten.

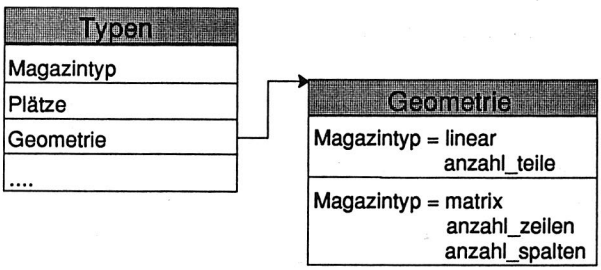


Bild 64: Magazintypen

Existiert für die Teileentnahme dann eine Berechnungsfunktion für die Koordinate des Greifpunktes (z.B. *ber_koord_greif (magazin)*), die bisher nur für Linear- und Matrix-Magazine eingesetzt wurde, so kann diese Funktion durch Erweiterung auch auf neue Magazintypen (z.B. kreisförmige) in einer Montagezelle übertragen werden. Hierzu muß zunächst ein neuer Magazintyp '*kreis*' eingeführt und für dessen Geometriebeschreibung die notwendigen CAD / Montageplanungsdaten übernommen werden (Bild 65). Die Entnahmefunktion als Bibliotheksmodul kann in der Implementierung um die Berechnungsfunktion "kreis_koord" erweitert werden, wird also mächtiger, hat aber für bisherige Anwender weiterhin das gleiche Erscheinungsbild.

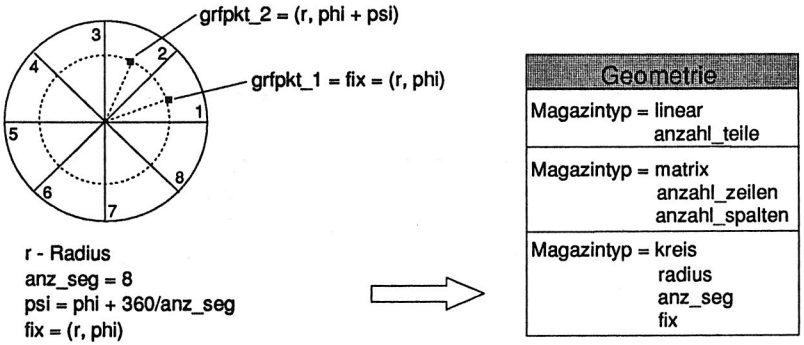


Bild 65: Kreismagazin mit Parametern

Zur Peripherieverwaltung können weiterhin unterschiedliche Algorithmen verwendet werden. Eine Umstellung der Verwaltung - Einführung der Suchstrategie mittels Bäumen (schnellere Suche und Vereinfachung von Ortswechseln bei zusammengesetzten Strukturen) anstatt Vektor-/Matrizensuche - erforderte für die Datenstrukturen (Magazin, Typen) eine Erweiterung um die Elemente *'zeiger_links'*, *'zeiger_rechts'*. Auf der Menge der Magazintypen ist eine Ordnung zu definieren und diese in einem sortierten Baum etc. zu verwalten. Eine Algorithmenauswahl zum Suchen (z.B. Inorder oder Postorder) kann dann entweder explizit an die Funktion *'such_geometrie (magazin)'* in einer Anwendung gebunden oder in die Funktion als zusätzlicher Parameter *'sort_art'* aufgenommen werden.

Wichtig ist neben der einheitlichen Modulstruktur die Erkenntnis über das Leistungsvermögen des Bausteines, d.h. zunächst ist das 'Wie wird die Leistung erbracht' nicht von Bedeutung. So können z.B. aufgrund von Laufzeitfragen, Genauigkeit, Speicherbedarf etc. mehrere Implementierungen für einen ADT existieren (Liste, binärer Baum). Bei der Konfiguration von Magazinstrukturen sind dann z.B. 4 Schritte zu beachten:

- vorhandene Module auf Eignung testen und evt. erweitern,
- Festlegung der eingesetzten Magazintypen und deren Eigenschaften (Füllen der Struktur *'typen'*),
- Bestimmung der Standorte (z.B. Füllen einer Struktur 'standorte' als Stammdatum),
- Zuordnung von Magazin zu Standort und Startbelegungen festlegen (Initialisierungsdaten z.B. für Struktur *'magazin'* und *'zuord_tabelle'*).

8 Zusammenfassung

Der Aspekt kostengünstiger und rationeller Softwareproduktion steht zunehmend im Mittelpunkt der Automatisierungstechnik. Besonders im Bereich der Variantenmontage sind zweckmäßige Steuerungsmaßnahmen gefragt, um nicht nur bemerkenswerte Einzellösungen zu konzipieren, sondern auch auf Basisentwicklungen zurückgreifen zu können. Ziel dieser Arbeit war es deshalb, innerhalb eines abgegrenzten Gebietes der Steurungstechnik diese Problematik aufzugreifen, ein Konzept zu entwickeln und durch Realisierungen zu erproben.

Zur Gestaltung von Softwaresystemen in rechnergeführten Montagezellen, d.h. zur Erfassung und Darstellung der auftretenden informationstechnischen Maßnahmen, wird ein methodischer Ansatz aufgezeigt. Das entwickelte Architekturmodell in Verbindung mit bestehenden Tools bildet die Basis für Struktur und Organisationsformen und für das Verständnis eines Gesamtgerüstes zum Aufbau von Zellenrechnersoftware.

Ausgehend von Flexibilitätsanforderungen und Entwicklungen innerhalb von Verfahrensketten in der Montageautomatisierung sowie der Integration bestehender technischer Systeme wurde der Denkansatz darauf abgestimmt, jeweilige Kernpunkte der konkreten Applikation auf die Ebenenstruktur übertragen und abbilden zu können.

Die Unterstützungsaspekte betreffen im technischen und organisatorischen Bereich nicht nur die aufbauorganisatorische, sondern vor allem auch die ablauforganisatorische Seite. Die angestrebte Strukturierung und Modularisierung sowie der Aufbau eines Baukastensystems erleichtern die Beschreibung von Schnittstellen und zeigen zugleich einen Ansatz, Standardsoftware und unterschiedliche Herstellerbausteine zu koppeln und eigene Konstrukte und Ablaufwünsche zu integrieren. Durch Aufzeigen von Aspekten möglicher Kommunikationsstrukturen untereinander und ereignisgesteuerter Abläufe wird erreicht, daß die Komplexität einer konkreten Applikation sich nicht allein auf das enge Gebiet der Individualentwicklung beschränkt.

Bezüglich Basissystemen sind Betriebssysteme wie UNIX und relationale Datenbanken wesentliche Komponenten, wobei Anforderungen an Realtimeeigenschaften und an dynamische Datenverteilung sicherlich noch weiterer Entwicklung bedürfen.

Ideen innerhalb der Funktionalität von prozeßnahen Softwaresystemen sind bekannterweise nur im Rahmen ihrer Vermittelbarkeit und technischen Machbarkeit realisierbar.

So bildet diese Modellierung, beruhend auch auf objektorientierter Darstellung und semantischer Beschreibung der montagezellenrelevanten Objekte die Voraussetzung zum effizienten Einsatz geeigneter Softwarewerkzeuge (CASE-Tools). Erst damit lassen sich erforderliche Schritte wie Aufgliederung von Steuerungssystemen, relationale Darstellung und Realisierung konkreter Anforderungen verbinden und in den einzelnen Phasen alle Beteiligten in den Entwicklungsablauf einbeziehen.

Mit Hilfe des Architekturmodells wurden somit Informations-, Funktionsstrukturen und Arbeitsabläufe aufeinander abgestimmt, wodurch unter Einbeziehung aktueller Daten stets die gewünschte Reaktionsflexibilität erhalten wird. Außerdem wurde gezeigt, wie Initialisierungs- und Fehlerbehebungsmechanismen in eine Softwarelösung zu integrieren sind. Für unterschiedliche Montagesteuerungsaufgaben wurden die Ergebnisse dieser Vorgehensweise aufgezeigt.

Die Realisierungen haben nicht nur die Machbarkeit und Effektivität bestätigt, sondern auch zur Transparenz und zur Übertragung von Teilaspekten beigetragen. Weiterhin kann man durch den hierarchischen Steuerungsaufbau und die Aufbereitung der parametrisierten Steuerdaten in den internen Regelkreisen jeweils die Problematik der Produktvielfalt in den Griff bekommen.

Das dargestellte systematische Vorgehen zeigte positive Auswirkungen auf den Softwareerstellungsprozeß und ermöglichte durch explizites Wissen über asynchrone Prozeßsysteme aus dem Bereich der Informatik und über projektspezifische Anforderungen aus der Fertigungswelt Synergieeffekte, auf die die Automatisierungstechnik nicht verzichten sollte und die das Teilgebiet 'Flexible Montagezellen mit Industrierobotern oder NC-Achsen' auch wirtschaftlich interessanter gestalten.

9 Literatur

1. Ahrens, W.; Polke, M.:
Informationsstrukturen in der Automatisierungstechnik. In: Mit vernetzten, intelligenten Komponenten zu leistungsfähigeren Mess- und Automatisierungskomponenten - INTERKAMA-Kongress 89. Oldenbourg Verlag, München Wien 1989

2. AMICE Consortium (Hrsg.):
CIM-OSA Reference Architecture Specifications. Brüssel 1988

3. Anderl, R. u.a.:
Integration von CA-Technologien - Ansätze für eine flexible Produktionstechnik; VDI-Bericht 20. VDI- Verlag, Düsseldorf 1987

4. AWV-AK4.2 AWV - Arbeitsgemeinschaft für wirtschaftliche Verwaltung e.V. (Hrsg.):
Softwarequalität: Beurteilungsmerkmale aus Anwendersicht. Eschborn 1987

5. Bärnreuther, B.; Scheller, J. u.a.:
Die Standardisierung von Datenschnittstellen in der Montageautomatisierung. Informatik-Spektrum 12 (1989), H. 6, S. 312-320

6. Bamberg, G.; Bauer, F.:
Statistik. 3. Auflage, Oldenbourg Verlag, München Wien 1984

7. Beier, H.H.:
Fertigungstechnik: Werkstattsteuerung mit neuem Profil. wt Werkstattstechnik 79 (1989), H. 8, S. 445-447

8. Bender, K. (Hrsg.):
Profibus - Der Feldbus für die Automation. Hanser Verlag, München Wien 1990

9. Bilger, B.:
Montage Praxis: Flexibel automatisieren. Resch Verlag, Gräfelfing 1987

10. Bullinger, H.J. (Hrsg.):
Systematische Montageplanung: Handbuch für die Praxis. Hanser Verlag, München Wien 1986

11. Bullinger, H.J.:
CIM - Die Herausforderung der nächsten Jahre in Forschung und Praxis. In: Produktionsforum '88 - Die CIM-fähige Fabrik; IPA-IAO Forschung und Praxis, T9. Hrsg. v. H.J. Bullinger. Springer Verlag, Berlin u.a. 1988

12. Classe, D.; Scholz, W.:
Verfahrensbeschreibende Parameter - Schlüssel zur systematischen Montageautomatisierung. wt Werkstatts-technik 77 (1987), H. 12, S. 684-689

13. Cox, I.J.; Gehani, N.H.:
Concurrent Programing and Robotics. Robotics Research 8 (1989), No. 2

14. DeMarco, T.:
Structured Analysis and System Specifications. Yordon Press, New York 1988

15. Dietsch, H.:
Feldbus. Informatik-Spektrum 13 (1990), H. 4

16. Dietsch, H.; Bärnreuther, B. u.a.:
Eine MMS-Implementierung in einer Feldbusumgebung; VDI-Berichte 723. VDI-Verlag, Düsseldorf 1989

17. Dillmann, R.:
Lernende Roboter: Aspekte maschinellen Lernens; Fachberichte Messen, Steuern, Regeln, 15. Springer Verlag, Berlin u.a. 1988

18. Deutsches Institut für Normung (Hrsg.):
Normung von Schnittstellen für die rechnerintegrierte Produktion (CIM): Standortbestimmung und Handlungsbedarf; DIN-Fachbericht, 15. Beuth Verlag, Berlin Köln 1987

19. Deutsches Institut für Normung (Hrsg.):
Schnittstellen der rechnerintegrierten Produktion (CIM): CAD und NC-Verfahrenskette; DIN-Fachbericht, 20. Beuth Verlag, Berlin Köln 1989

20. Deutsches Institut für Normung (Hrsg.):
Schnittstellen der rechnerintegrierten Produktion (CIM): Fertigungssteuerung und Auftragsabwicklung. DIN-Fachbericht, 21. Beuth Verlag, Berlin Köln 1989

21. Duelen, G.; Linnemann, H.; Bernhardt, R.:
Die Informationsarchitektur in datengetriebenen Fabriken. In: Tagungsband zum Produktionstechnischen Kolloquium. Berlin 1986

22. Dungern, O.v.; Freyberger, F.; Schmidt, G.:
Eine modulare Grundsteuerung für flexible Montagezellen. atp Automatisierungstechnische Praxis 32 (1990), H. 1, S. 22-29

23. Eisele, R.:
Konzeption und Wirtschaftlichkeit rechnerintegrierter Planungssysteme. Diss. Univ. Erlangen-Nürnberg; Hanser Verlag, München Wien 1990

24. Emulex corporation (Hrsg.):
DCP/MUX Installation and Technical Reference Guide. 1986

25. Endres, A.:
Softwarewiederverwendung, Ziele, Wege und Erfahrungen. Informatik-Spektrum 11 (1988), H. 2

26. Endres, A.:
Einige Grundprobleme der Software-Wiederverwendung und deren Lösungsmöglichkeiten. In: Softwaretechnik in Automatisierung und Kommunikation; ITG-Fachbericht, 109. VDE Verlag, Berlin Offenbach 1989

27. Essler, W.K.:
Die Sprache der Logik. In: Grundprobleme der großen Philosophen. Hrsg. v. J. Speck. Vandenhoeck & Ruprecht Verlag, Göttingen 1979

28. Eversheim, W.; Fischer, W.; Steudel, M.:
Modulare Systemvarianten zur automatischen Arbeitsplanerstellung; Forschungsbericht des Landes Nordrhein-Westfalen, Nr. 2948. Westdeutscher Verlag, 1980

29. Feldmann, K.; Schmidt B. (Hrsg.):
Simulation in der Fertigungstechnik. Springer Verlag, Berlin u.a. 1988

30. Feldmann, K.:
Rechnerintegrierte Montagesysteme; Fachtagung 'Rechnerintegrierte Produktionssysteme'. Erlangen 1987, S.279-294.

31. Feldmann, K.:
Materialfluß und Informationsverarbeitung - Basis rationeller Montageautomatisierung; VDI-Berichte 871. VDI-Verlag, Düsseldorf 1990

32. Firnau, J.:
Flexible Fertigungssysteme - Entwicklung und Erprobung eines zentralen Steuerungssystems. Springer Verlag, Berlin u.a. 1982

33. Fischer, H.:
Verteilte Planungssysteme zur Flexibilitätssteigerung der rechnerintegrierten Teilefertigung. Diss. Univ. Erlangen-Nürnberg; Hanser Verlag, München Wien 1990

34. Fischer, K.:
Regelbasierte Synchronisation von Robotern und Maschinen in der Fertigung. Interner Bericht, Mathematisches Institut und Institut für Informatik der Technischen Univ. München, 1988

35. Förster, H.-U.; Hiert, K.:
Entwicklung von Anforderungsprofilen flexibler automatisierter Fertigungskonzepte an die Produktionsplanung und -steuerung; Abschlußbericht zum Forschungsvorhaben Nr. 5 134. FIR, Aachen 1987

36. Frommherz, G.; Werker G.:
Spezifikation von dreidimensionalen Szenen durch graphische Definition räumlicher Beziehungen. Robotersys-teme 1989, H. 5, S. 197-208

37. Fu, K.S.; Gonzalez, R.C.; Lee, C.S.G.:
Robotics: Control, Sensing, Vision, and Intelligence. McGraw-Hill Book Company, New York 1987

38. Ganderon, E.:
Dezentrale Organisationsstrukturen der Produktion. In: Produktionsforum '88 - Die CIM-fähige Fabrik; IPA-IAO Forschung und Praxis, T9. Hrsg. v. H.-J. Bullinger. Springer Verlag, Berlin u.a. 1988

39. Gautier, M.:
Real-time Implementation of Dynamic control of Robot. In: IFAC Proceedings of Robot Control 1988. Pergamon Press, Oxford 1989

40. Geitner, U.W. (Hrsg.):
CIM-Handbuch: Wirtschaftlichkeit durch Integration. Vieweg Verlag, Braunschweig Wiesbaden 1987

41. Grabowski, H.; Glatz, R.:
Schnittstellen zum Austausch produktdefinierender Daten. VDI-Z 128 (1986), H. 10

42. Grabowski, H.; Anderl, R. u.a:
STEP - Entwicklung einer Schnittstelle zum Produktaustausch. VDI-Z 131 (1989), H. 9, S. 68-76

43. Härder, Th.; Reuter, A.:
Architektur von Datenbanksystemen für Non-Standard Anwendungen. In: Datenbanksysteme für Büro, Technik und Wissenschaft; Informatik-Fachberichte, 94. Springer Verlag, Berlin u.a. 1985

44. Hallmann, M.:
Prototyping komplexer Softwaresysteme. Teubner Verlag, Stuttgart 1990

45. Heinrich, L.J.; Roithmayr, F.:
Wirtschaftsinformatik-Lexikon. Oldenbourg Verlag, München u.a. 1986

46. Hesse, W.:
Software-Qualitätssicherung. Informatik-Spektrum 10 (1987)

47. Hörmann, A.; Hugel, T.; Meier, W.:
Ein Ansatz zur Realisierung intelligenter fehlertoleranter Robotersysteme. Robotersysteme 4 (1988), H. 4, S. 223-231

48. Hofmann, W.:
Koordinierungsprobleme und ihre Implementierung auf aktuellen Multiprozessorsystemen. In: SFB 182 Arbeitsbericht 89/5. Univ. Erlangen-Nürnberg, 1989

49. Hofmann, F.:
Betriebssysteme: Grundkonzepte und Modellvorstellungen. Teubner Verlag, Stuttgart 1984

50. Ish-Shalom, J.; Kazanzides, P.:
SPARTA: Multiple signal processors for high-performance robotic control. In: Proceedings of 1988 IEEE International Conference on Robotics and Automation; Vol. 1. Computer Society Press, Washington 1988, S. 284-290

51. ISO IS 9506: Manufacturing Message Specification;Part1.
Service Definition, Part 2: Protokoll Specification. 1988

52. Jablonski, S.:
Datenverwaltung in verteilten Systemen: Grundlagen und Lösungskonzepte. Springer Verlag, Berlin u.a. 1990

53. Jalote, P.:
Functional Refinement and Nested Objects for Objects-Oriented Design. IEEE Transactions on Software Engineering Vol. 15 (1989), No. 3, pp. 264-270

54. Keramidis, S.:
Eine Methode zur Spezifikation und korrekten Implementierung von asynchronen Systemen; Arbeitsberichte des Instituts f. Math Maschinen u. Datenverarbeitung, Universität Erlangen-Nürnberg, Bd. 15, Nr. 4. Erlangen 1982

55. Klotzbücher, R.; Scheller, J.:
Steuerungssystem für Zellenrechner zur Lösung von Flexibilitätsanforderungen in der Montageautomatisierung. dima Die Maschine 1988, H. 7/8, S. 65-68 und H. 10, S. 18-22

56. Kohen, E.:
Adaptierbare Steuerungssoftware für flexible Fertigungssysteme - Ein Beitrag zur Reduzierung der Softwarekosten in der Fertigungstechnik. Diss. TH Aachen, 1986

57. Komischke, M.:
Einbindung von Sensorsystemen in den Informationsfluß integrierter Produktionssysteme; Fortschr.-Berichte VDI, Reihe 8, Nr. 162. VDI Verlag, Düsseldorf 1988

58. Konz, H.J.:
Steuerung der Standplatzmontage komplexer Produkte. Diss. TH Aachen, 1989

59. Kurbel, K.:
Software Engineering im Produktionsbereich. Verlag Gabler, Wiesbaden 1983

60. Lautner, J.:
Untersuchung der Datenschnittstelle zwischen Zellensteuerung und operativer Ebene beim Montieren. Dipl.-Arb. Univ. Erlangen-Nürnberg, 1989

61. Limmer, R.:
Relationale Datenmodelle, eine Basis für die Generierung von Steuerdaten in einer flexiblen Montagezelle. Dipl.-Arb. Univ. Erlangen-Nürnberg, 1989

62. Lotter, B.:
Wirtschaftliche Montage: Ein Handbuch für Elektrogerätetechnik, Bau und Feinwerktechnik. VDI Verlag, Düsseldorf 1986

63. Maaß, S.; Oberquelle, H.:
Software-Ergonomie '89: Aufgabenorientierte Systemgestaltung und Funktionalität. Teubner Verlag, Stuttgart 1989

64. Mackert, L.:
Modellierung, Spezifikation und korrekte Realisierung von asynchronen Systemen; Arbeitsbericht des Instituts f. Math. Maschinen u. Datenverarbeitung, Universität Erlangen-Nürnberg, Bd.16, Nr. 7, Juli 1983

65. Maier, U.:
Arbeitsgangterminierung mit variablen strukturierten Arbeitsplänen: Ein Beitrag zur Fertigungssteuerung flexibler Fertigungssysteme. Springer Verlag, Berlin u.a. 1980

66. Mertens, P.:
Industrielle Datenverarbeitung; Band 1: Administrations- und Dispositionssysteme. 7. Auflage, Verlag Gabler, Wiesbaden 1988

67. Mertins, K.:
Steuerung rechnergeführter Fertigungssysteme; Produktionstechnik-Berlin, 37. Hanser Verlag, München Wien 1985

68. Meyer, M.; Hansen, K.:
Planungsverfahren des Operation Research; Band 1 u. Band 2. Verlag Girardet, Essen 1979

69. Milberg, J.; Groha, A.:
Der Zellengedanke als Strukturierungsprinzip im Informations- und Materialfluß flexibler Fertigungssysteme. ZwF CIM 81 (1986), H. 12, S. 682-686

70. Neighbors, J.M.:
The Draco Approach to Constructing Software from Reuseable Components. IEEE Transactions on Software Engineering Vol. SE-10 (1984), No. 5, pp. 564-574

71. Noltemeier, H.:
Graphen, Algorithmen, Datenstrukturen; Workshop: Graphentheoretische Konzepte. Hanser Verlag, München Wien 1976

72. Ortner, E.; Söllner, B.:
Semantische Datenmodellierung nach der Objekttypenmethode. Informatik-Spektrum 12 (1989), H. 1

73. Panse, R.:
CIM-OSA - Ein herstellerunabhängiges Unternehmensmodell. DIN-Mitt. 69 (1990), Nr. 3, S. 156-164

74. Parnas, D.L.:
On the Design and Development of Program Families, IEEE Transactions on Software Engineering Vol. 2 (1976), No. 1, pp. 1-9

75. Parthier, U.:
Relationales Modell, unternehmensweit für CAD/CAM/CIM. Hard & Software 1987, H. 5

76. Pepper, P.:
Neue Ansätze zur Wiederverwendbarkeit von Software. In: Softwaretechnik in Automatisierung und Kommunikation; ITG-Fachbericht, 109. VDE Verlag, Berlin Offenbach 1989

77. Perl, J.:
Graphentheorie: Grundlagen und Anwendung. Akademische Verlagsgesellschaft, Wiesbaden 1981

78. Prack, K.-W.:
Systemkonzept zur standardisierten rechnerunterstützten Arbeitsplanung; Fortschritt-Berichte VDI, Reihe 2 Nr. 100. VDI Verlag, Düsseldorf 1985

79. Pritschow, G.:
Die flexible Fertigungszelle - Chance und Herausforderung auch für den mittelständischen Betrieb. wt Werkstattstechnik 75 (1985) H. 11, S. 663-668

80. Pritschow, G.; Spur, G.; Weck, M.:
Sensordatenverarbeitung in der Fertigungstechnik. Hanser Verlag, München 1987

81. Rasmussen, J.:
The Role of Hierarchical Knowledge Representation in Decisionmaking and Systemmanagement. IEEE Transactions on Systems, Man and Cybernetics Vol. SMC-15 (1985), No. 2, pp. 234-243

82. Ratcliff, B.:
Software Engineering: Principles and Methods. Blackwell Scientific Publications, Oxford London Edinburgh 1987

83. Reichel, H.:
Nutzung der Flexibilität eines FFS und ihre Auswirkungen auf die Werkzeugversorgung. dima Die Maschine 1990, H. 3, S. 30-34

84. Reisig, W.:
Systementwurf mit Netzen. Springer Verlag, Berlin u.a. 1985

85. Rembold, U. u.a. (Hrsg.):
CAM-Handbuch. Springer-Verlag, Berlin u.a. 1990

86. Reuter, A.:
Fehlerbehandlung in Datenbanksystemen. Hanser Verlag, München Wien 1981

87. Roschmann, K.:
Betriebsdatenerfassung in Industrieunternehmen. Verlag Moderne Industrie, München 1979

88. Scheer, A.-W.:
CIM Computer Integrated Manufacturing: Der computergesteuerte Industriebetrieb. Springer Verlag, Berlin u.a. 1987

89. Scheer A.-W.:
CIM Aktivitäten in den USA. In: Genormte Schnittstellen für CIM. Hrsg. v. Deutsches Institut für Normung. Beuth Verlag, Berlin Köln 1989

90. Scheller J.:
Aspekte der Betriebsdatenerfassung in flexiblen Montagezellen. dima Die Maschine 1989, H. 9, S. 72-80

91. Scheller, J., Sommer, E.:
Hierarchisches Steuerungskonzept für flexible Montagezellen. atp Automatisierungstechnische Praxis 31 (1989), H. 4, S. 166-173

92. Scheller, J., Zeis, G.:
Wege zur rationellen Softwareerstellung für die Montageautomatisierung. atp Automatisierungstechnische Praxis 32 (1990), H. 12, S. 613-618

93. Scheschonk, G.:
Eine auf Petri-Netzen basierende Konstruktionsanalyse und (Teil-)Verifikationsmethode zur Modellierungsunterstützung bei Entwicklung von Informationssystemen. Diss. Techn. Univ. Berlin, 1984

94. Scholz, B.:
CIM-Schnittstellen: Konzepte, Standards und Probleme der Verknüpfung von Systemkomponenten in der rechnerintegrierten Produktion. Oldenbourg Verlag, München Wien 1988

95. Scholz, W.:
Modell zur datenbankgestützten Planung automatisierter Montageanlagen. Diss. Univ. Erlangen-Nürnberg; Hanser Verlag, München Wien 1989

96. Schuhmann,J.; Gerisch, M.:
Softwareentwurf - Prinzipien, Methoden, Arbeitsschritte, Rechnerunterstützung. Verlagsgesellschaft R. Müller, Köln 1986

97. Schwarz, K.
Manufacturing Message Specification (MMS) - Offene Verständigung in verteilten Systemen der industriellen Automatisierung. atp Automatisierungstechnische Praxis 31 (1989), H. 1

98. Severin, F.:
Planung der Flexibilität von roboterintegrierten Bearbeitungs- und Montagezellen. Hanser Verlag, München Wien 1987

99. Siemens AG (Hrsg.):
Robot Control Machine, RCM 2, Kopplung zum visuellen Sensor.
Gerätehandbuch Kommunikationsprozessor CP525.
Gerätehandbuch Mobiles Datenspeicherungssystem MOBY-M.

100. Sommer, E.:
Anforderungen an die Parallelverarbeitung bei Steuerung von Montagegeräten. In: SFB 182 Arbeitsbericht 90/1. Univ. Erlangen-Nürnberg, 1990

101. Sperber, M.:
Steuerungssystem für Handhabungs- und Fügevorgänge in einer flexiblen Montagezelle. Dipl.-Arb. Univ. Erlangen-Nünrberg, 1987

102. Spitz, P.:
Logistik zur Initialisierung von flexiblen Montagezellen. Dipl.-Arb. Univ. Erlangen-Nürnberg, 1989

103. Spitzner, W.:
Qualitätskosten und ihre Bewertung für das Unternehmen. In: Tagungsunterlagen zum Qualitätssicherungs-Kolloquium des TÜV-Rheinland. Verlag TÜV-Rheinland, Köln 1987

104. Spur, G.; Stöferle, T. (Hrsg.):
Handbuch der Fertigungstechnik; Band 5: Fügen, Handhaben, Montieren. Hanser Verlag, München Wien 1986

105. Steinbauer, D.; Wedekind, H.:
Integritätsaspekte in Datenbanksystemen. Informatik-Spektrum 8 (1985), H. 8, S. 60-68

106. Steusloff, H.:
Systemengineering für industrielle Automation. atp Automatisierungstechnische Praxis 32 (1990), H. 3, S. 129

107. Vardabedian, A.:
Report on a MMS Production Management Companian Standard (PMCS). August 1988

108. Verein Dt. Ingenieure (Hrsg.):
Lexikon der Produktionsplanung und -steuerung; VDI-Taschenbuch, T77. VDI Verlag, Düsseldorf 1983

109. Vetter, M.:
Strategie der Anwendungssoftware-Entwicklung. Teubner Verlag, Stuttgart 1988

110. Vits, R.:
Externes Werkzeughandling - Die Abhängigkeiten vom internen Werkzeugwechsel und Produktmix müssen erkannt werden. Maschinen und Methoden 1987, H.6

111. Wedekind, H.:
Datenorganisation. Verlag de Gryter, Berlin 1975

112. Wedekind, H.:
Technische Datenbanken; Datenbanken 1 und 2. Vorlesungsskripte, Univ. Erlangen-Nürnberg, 1987

113. Wiendahl, H.-P.:
Belastungsorientierte Fertigungssteuerung. Hanser Verlag, München Wien 1987

114. Wiendahl, H.-P., Ullmann, W.:
Anforderungen an die Bereitstellungsplanung von Werkzeugen in einer Fertigungssteuerung. Werkstatt und Betrieb 121 (1988), H. 2, S. 133-136

115. Willmer, H.:
Systematische Software-Qualitätssicherung anhand von Qualitäts- und Produktmodellen. Springer Verlag, Berlin u.a. 1985

116. Zeis, G.; Scheller, J.:
Zellenrechner in der Montage, Wege zur rationellen Softwareerstellung. Interner Bericht Nr. 12 IMMD 4, Univ. Erlangen-Nürnberg, 1989

117. Zörntlein, G.:
Flexible Fertigungssyteme: Belegung, Steuerung, Datenorganisation. Diss. Univ. Erlangen-Nürnberg; Hanser Verlag, München Wien 1987

LEBENSLAUF

Persönliches:	
Name:	Josef Scheller
Geboren:	am 28. Januar 1958 in Gerolzhofen
Eltern:	Adolf Scheller Hedwig Scheller, geb. Issing
Familienstand:	Verheiratet mit Andrea Scheller, geb. Fiedler zwei Söhne, Dominik und Fabian Scheller
Staatsangehörigkeit:	deutsch
Schulbildung:	
1964 - 1967	Grundschule Brünnstadt
1967 - 1969	Verbandschule Frankenwinheim - Brünnstadt
1969 - 1978	Humanistisch-neusprachliches Gymnasium Gaibach
29.06.1978	Abitur
Wehrdienst:	
1978 - 1979	Grundwehrdienst in Panzergrenadiereinheit, Mellrichstadt und Fernmeldeeinheit, Veitshöchheim
Studium:	
1979 - 1985	Friedrich Alexander Universität Erlangen/Nürnberg Studienfach: Informatik Schwerpunkt: Betriebssysteme Nebenfach: Betriebswirtschaft
28.09.1985	Diplomhauptprüfung
Berufstätigkeiten:	
1982 - 1985	Praktikantentätigkeiten in Industrieunternehmen und studentische Hilfskraft am Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik
1985 - 1986	Mitarbeiter der Siemens AG - Bereich Medizinische Technik in Erlangen
1986 - 1991	Wissenschaftlicher Mitarbeiter am Lehrstuhl für Fertigungsautomatisierung und Produktionssystematik der Universität Erlangen - Nürnberg

Reihe Fertigungstechnik Erlangen

Band 1
Andreas Hemberger
Innovationspotentiale in der rechnerintegrierten Produktion durch wissensbasierte Systeme
208 Seiten, 107 Bilder. 1988. Kartoniert.

Band 2
Detlef Classe
Beitrag zur Steigerung der Flexibilität automatisierter Montagesysteme durch Sensorintegration und erweiterte Steuerungskonzepte
194 Seiten, 70 Bilder. 1988. Kartoniert.

Band 3
Friedrich-Wilhelm Nolting
Projektierung von Montagesystemen
201 Seiten, 107 Bilder, 1 Tabelle. 1989.
Kartoniert.

Band 4
Karsten Schlüter
Nutzungsgradsteigerung von Montagesystemen durch den Einsatz der Simulationstechnik
177 Seiten, 97 Bilder. 1989. Kartoniert.

Band 5
Shir-Kuan Lin
Aufbau von Modellen zur Lageregelung von Industrierobotern
168 Seiten, 46 Bilder. 1989. Kartoniert.

Band 6
Rudolf Nuss
Untersuchungen zur Bearbeitungsqualität im Fertigungssystem Laserstrahlschneiden
206 Seiten, 115 Bilder, 6 Tabellen. 1989. Kartoniert.

Band 7
Wolfgang Scholz
Modell zur datenbankgestützten Planung automatisierter Montageanlagen
194 Seiten, 89 Bilder. 1989. Kartoniert.

Band 8
Hans-Jürgen Wißmeier
Beitrag zur Beurteilung des Bruchverhaltens von Hartmetall-Fließpreßmatrizen
179 Seiten, 99 Bilder, 9 Tabellen. 1989. Kartoniert.

Band 9
Rainer Eisele
Konzeption und Wirtschaftlichkeit von Planungssystemen in der Produktion
183 Seiten, 86 Bilder. 1990. Kartoniert.

Band 10
Rolf Pfeiffer
Technologisch orientierte Montageplanung am Beispiel der Schraubtechnik
216 Seiten, 102 Bilder, 16 Tabellen. 1990. Kartoniert.

Band 11
Herbert Fischer
Verteilte Planungssysteme zur Flexibilitätssteigerung der rechnerintegrierten Teilefertigung
201 Seiten, 82 Bilder. 1990. Kartoniert.

Band 12
Gerhard Kleineidam
CAD/CAP : Rechnergestützte Montagefeinplanung
203 Seiten, 107 Bilder. 1990. Kartoniert.

Band 13
Frank Vollertsen
Pulvermetallurgische Verarbeitung eines übereutektoiden verschleißfesten Stahls
XIII + 217 Seiten, 67 Bilder, 34 Tabellen. 1990. Kartoniert.

Band 14
Stephan Biermann
Untersuchungen zur Anlagen- und Prozeßdiagnostik für das Schneiden mit CO_2 - Hochleistungslasern
VIII + 170 Seiten, 93 Bilder, 4 Tabellen. 1991. Kartoniert.

Band 15
Uwe Geißler
Material- und Datenfluß in einer flexiblen Blechbearbeitungszelle
124 Seiten, 41 Bilder, 7 Tabellen. 1991. Kartoniert.

Band 16
Frank Oswald Hake
Entwicklung eines rechnergestützten Diagnosesystems für automatisierte Montagezellen
XIV + 166 Seiten, 77 Bilder. 1991. Kartoniert.

Band 17
Herbert Reichel
Optimierung der Werkzeugbereitstellung durch rechnergestützte Arbeitsfolgenbestimmung
198 Seiten, 73 Bilder, 2 Tabellen. 1991. Kartoniert.

Band 18
Josef Scheller
Modellierung und Einsatz von Softwaresystemen für rechnergeführte Montagezellen
198 Seiten, 65 Bilder. 1991. Kartoniert.