

Prozesssteuerung an der Sektion Informationsverarbeitung der TU Dresden

Forschung, Überführung in die Praxis und Ausbildung

ERWIN SCHMIDT

Erwin.Schmidt@tu-dresden.de

An der 1969 gegründeten Sektion Informationsverarbeitung der TU Dresden wurde im Bereich von Prof. Dr.-Ing. Heinz Stahn zur Steuerung diskreter Prozesse mit Anwendungsziel Maschinenbau geforscht. Wegen geringer Kapazitäten für die Automatisierung in allen Bereichen bestand die Chance und gleichzeitig die Notwendigkeit, die Ergebnisse unter Mitwirkung der Firmen in die Praxis zu überführen und dort zu überprüfen. Nach Entwicklungsarbeiten zu einer Robotersteuerung wurden Einsatzvarianten derselben bearbeitet. Parallel dazu entwickelte Dr.-Ing. Martin Engelen eine Allgemeine und Rekursive Strukturierung zur Algorithmenprojektierung [2], die dabei und in Fertigungssteuerungsprozessen im Bereich CAM angewendet und verifiziert wurde [3]. Die Rekursion erfolgt unter Zuhilfenahme fachbezogener heuristischer Methoden, deren Fehlen bei Anwendung in der Studentenausbildung zu unterschiedlichen Resultaten führen kann. Mit der Verbesserung der technischen Möglichkeiten konnten größere Vorhaben bedienerfreundlicher realisiert werden.

1 Robotersteuerung IRS 650

Im VEB NUMERIK Karl-Marx-Stadt wurde eine Steuerung auf Basis von 8-bit-Prozessoren und einer NC-Sprache [5] vorwiegend für den Einsatz im Maschinenbau mit Gelenkrobotern des Kombinats Fortschritt entwickelt, für welche wir das Satzbetriebssystem (SABE) erstellten.

1.1 Satzbetriebssystem IRS 650

Das Satzbetriebssystem (SABE) wurde von einer von Numerik vorgegebenen Zentralen Ablaufsteuerung aufgerufen und nutzte deren Achssteuerung, E/A-Steuerung (je 48 Stück) und Bedienblendensteuerung. Es standen 11 kB EPROM und 0,5 kB RAM als Speicher zur Verfügung.

Zur Steuerung der sechs Betriebsarten wurde je nach Stellung eines Betriebsart-Wahlschalters von einem Rahmen-Steuermodul auf entsprechende

Betriebsart-Steuermodule verzweigt. Die Funktionen der Module wurden mittels Zustandsgraphen modelliert. Zur Abarbeitung stellte der Rahmen-Steuermodul einen „Verteiler“ bereit, eine frühe Form eines Zustandsgraphen-Interpreters. Entsprechend dem Wert der Zustandsgröße verzweigte dieser in das jeweils aktuelle Zustands-Unterprogramm, in dem die zulässigen Zustandsübergänge behandelt wurden.

Die Programmierung erfolgte in NC-Sprache entweder textuell (durch Technologen oder per direkter Eingabe an der Bedienblende) oder an der Steuerung mittels indirektem Teach-in über die Bedienblende. Eine Kombination beider Methoden war möglich durch die Änderung vorgefertigter Sätze während des Testlaufes nach den Methoden des „hot editing“. Da die Steuerung intern nur mit einem Objektcode arbeitete, bekam das SABE ein Übersetzersystem von der NC-Sprache zur Ausführung und ein Rückübersetzersystem für die Anzeige und Ausgabe. Ein Zwischencode gestattete eine effektive Implementierung.

Für die Programmierung von Gelenkrobotern in kartesischen Koordinaten entwickelte die Sektion Mathematik der TU eine sehr schnell und genau arbeitende Koordinatentransformation.

1.2 Bedienalgorithmen

Tabelle 1: BAUTO (Auszug) B-blinkend, L-leuchtend, ZL-Ziffern leuchtend, „X“-Taste X

Nr.	Bedingung Anzeige	Bedienwunsch	Aktionen	Sprung zu	Kommentar
1.1.	B: % ,L,N	Programm anstellen	„%“ (oder „L“)	3.1.	(zum Test evtl. auch UP)
1.4.	„	BA beenden	-	BRAHM	Zu BA-Wahl
3.1.	L: N	Programm-Nr. eingeben, beenden	„ZI“... -> „BE“	3.2.	1 oder 2 Ziffern
3.2.	L: N ZL:Satznr.	Nr. ändern	„ZI“ ...	3.3.	Vorschlag 1. Satznr.
3.3.	„	Wahl beenden	„BE“	4.1.	
4.1.	B: START , SCHR,EF	Automatiklauf	„START“	5.1.	Routinebetrieb

Ein Bildschirm wie an Numerischen Bearbeitungszentren war nicht vorhanden. Zur Bedienerführung wurden LEDs der Tasten und die Ziffernanzeige leuchtend oder blinkend geschaltet. Dann konnte dem Bediener eine Darstellung in

Form einer Entscheidungstabelle mit allen, insbesondere auch selten zulässigen, Handlungen gegeben werden. Tabelle 1 zeigt das Prinzip in einem Auszug, in Quelle [5] ist der Bedienalgorithmus für das Programmieren angegeben. Zusätzlich wurden Standard-Handlungsfolgen zur ersten Einarbeitung in die Bedienalgorithmen und andererseits Graphen-Darstellungen unter Weglassung der Aktionen und Kommentare für erfahrene Bediener abgeleitet.

1.3 Einsatz bei Zellensteuerungen im Kombinat Fortschritt [8]

Stufenlose Getriebe (Variatoren) sollten in zwei Zellen montiert werden. Die Pressen P1/P2 und die Schraubstation SS besaßen Eigenbau-Steuerungen, die Roboter die IRS 650. Weiterhin waren Spanntisch ST, Kleinteilelager KT und Rollenbahnen 6-13 vorhanden. Jede Zelle verfügte über eine Speicherprogrammierbare Steuerung MRS 700, auf der eine relativ autonome Zellensteuerung die Stationssteuerungen entsprechend des Arbeitsplans für einen von 11 Variatortypen koordinierte. Die Kopplung mit den Stations-Steuerungen, also auch mit der IRS wurde über digitale Ein- und Ausgänge realisiert. Nach Starten des jeweiligen Roboterprogramms wurden z.B. in der Presszelle Führungsbuchsen mit mehreren Kleinteilen in Presse 1 komplettiert und auf Presse 2 Scheibenhälften vormontiert, bevor sie nach Rollenbahntransport in der Schraubzelle zusammengeschaubt werden konnten.

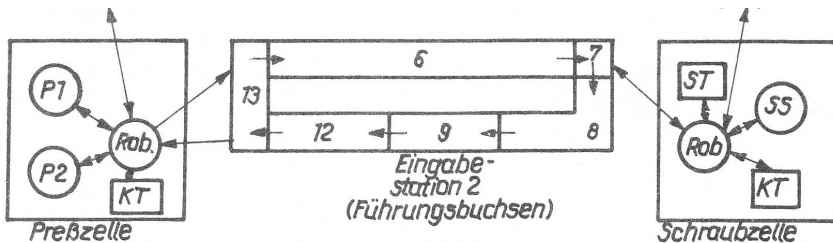


Abbildung 1: Ausschnitt Variatormontage: Zellen

1.4 Einsatz an der TU Dresden

Anfang des Jahres 1986 konnten wir schließlich ein Funktionsmuster der Steuerung mit einem 10-kg-Gelenkroboter des Kombinats Fortschritt am Lehrstuhl von Prof. Zachau an der Fakultät Maschinenwesen als „Intersektionelle Zusammenarbeit“ im Beisein des Rektors in Betrieb nehmen.

Wegen Schwerpunktverlagerung nach der Gründung des Informatik-Zentrums im Herbst 1986 ist der beabsichtigte Einsatz bei der Ausbildung der Informatik-Studenten allerdings nicht realisiert worden.

2 CAMARS-Technologie [3]

Computer Aided Manufacturing Systems (CAM-Systeme) besitzen lokale Steuerungen auf Stationsebene und benötigen übergeordnete Koordinations-ebenen. Eine Menge von Technologien (Operationslisten) der zu bearbeitenden Werkstücke bildet den aktuell zu realisierenden Prozess. Dieser kann mittels Zustandsgraphen zusammen mit Petrinetz-Regeln realisiert werden. Aus diesen Vorgaben kann man die Entwicklung der Software für Stations-, Zellen-, Abschnitts- und Bereichssteuerungen ableiten und die Kommunikation zwischen diesen. Als „Bauplan“ entsteht ein Zwischenprodukt Prozess-Struktur, das zur Erhöhung der Softwarequalität beiträgt und die Zusammenarbeit mit Vertretern verschiedener Fachgebiete gestattet.

Diese Strukturierung einer Aufgabe ist zwar nicht algorithmierbar, aber mit den Methoden der CAMARS-Technologie heuristisch mit Fachkenntnissen lösbar.

2.1 Grundlagen

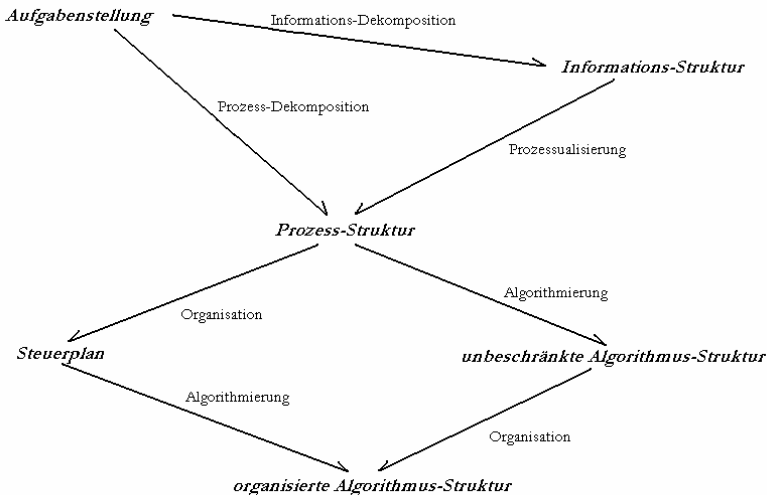


Abbildung 2: Übersichtsschema Allgemeine Rekursive Strukturierung

Aus der Aufgabenstellung für den Gesamtprozess kann man entweder von den physischen Komponenten des Basissystems die Steuerungs-Teilprozesse ableiten oder es gelingt, durch Einführung von Zwischen-Informationen eine Struktur auf diesen Informationen zu erzeugen. Da jede Information in einer Operation entstehen muss, kann man auch daraus eine Prozess-Struktur gewinnen.

Aus der Prozess-Struktur heraus kann dann entweder ein unbeschränkter Algorithmus mit allen parallelen Vorgängen entwickelt werden, der anschließend entsprechend den zur Verfügung stehenden Prozessoren eingeschränkt/organisiert wird. Oder man organisiert z.B. zuerst eine Bearbeitungsfolge auf einem Einprozessorsystem und algorithmiert dann. Als Programmiersprache verwendeten wir C. Nach Erstellung von Prototypen können diese dann rekursiv verbessert werden. Außerdem gestattet dies eine Testung vor Fertigstellung aller Teilalgorithmen.

Bezeichnet man die informationellen Operanden als Entitäten, so kann man eine weitere Forderung formulieren: „Die Struktur sollte so weit verfeinert werden, dass die Entitäten mittels Zustandsgraphen beschrieben werden können.“ Im funktional-logischen Entwurf sollten noch keine implementationspezifischen Elemente enthalten sein, was den Austausch mit Fachleuten anderer Richtungen gestattet. Der Implementationsentwurf kombiniert diesen dann z.B. mit Festlegungen zu Datenstrukturen, Namen von Bedingungs- und Aktionsroutinen und deren programmtechnischem Entwurf. Die Zustandsgraphen können dabei erweitert werden, dürfen den funktional-logischen aber nicht widersprechen. Mit einem Aufbereitungsprogramm lassen sich die Datenstrukturen für einen Zustandsgrapheninterpreter erzeugen. Der Programmierer implementiert dann die Bedingungen und Aktionen als elementare Funktionen sehr einfachen Aufbaus. Der Interpreter SIRIUS 2.2 [1] bearbeitet ereignisorientiert alle aktivierten Entitäten so lange, bis keine Aktivierung mehr vorliegt. Bei Selbstaktivierung einer oder mehrerer Entitäten wird SIRIUS nie verlassen. Im unten beschriebenen FMCS 2500 hatte der Interpreter zeitweise über 2.000 Entitäten quasiparallel in Echtzeit hantiert.

2.2 Einsatz in der Lehre

Als Beispiele dienen in der Lehre die Steuerung einer Bohrmaschine, einer Fräsmaschine, eines Handarbeitsplatzes, eines Drehtisches und deren Kombination an einem Rundschalttisch.

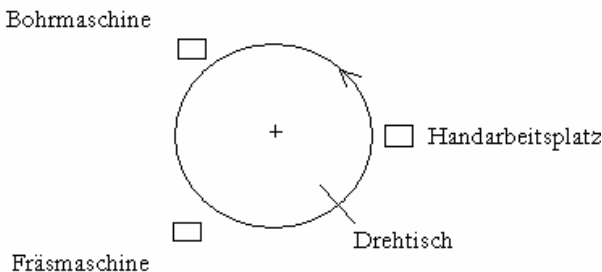


Abbildung 3: Schema Taktstraße

An einem Handarbeitsplatz werden Teile aufgespannt und nach Durchlauf wieder entfernt. Die Maschinen sollten Antriebe mit Drehrichtungsumkehr und zwei Endschaltern, der Drehtisch nur Einrichtungsantrieb und einen Endschalter besitzen, der Handarbeitsplatz Lampen für die Signalisierung an den Bediener und Taster für die Endmeldung.

Für die Gesamtsteuerung konnte eine Struktur entworfen werden, bei der die Steuerung des Tisches erweitert wurde um die Synchronisation mit den Stationssteuerungen. War die Steuerung des Tisches wegen Racing schon allein schwierig zu verstehen, so wurde es mit der Erweiterung noch schwerer. Als Alternative wurde eine Struktur mit einer den Stationssteuerungen übergeordneten Koordination bearbeitet, in der dann auch die Kommunikation mit der Umgebung angeordnet werden konnte, ohne die Einzelsteuerungen wesentlich zu ändern.

Wegen des fehlenden fachlichen Hintergrundes mussten den Studenten aber Hilfestellungen insbesondere zur Koordination gegeben werden.

3 Flexibles Fertigungssteuerungssystem FMCS 2500 bei Mikromat Dresden [4, 7]

Nach der Bearbeitung eines Fertigungssteuerungssystems FMS 2200 bei UNION Gera realisierten wir in Zusammenarbeit mit dem Betrieb und mit großem Einsatz von Studenten ein Steuerungssystem in Dresden aus vier Bearbeitungszentren (CFZ, CFBKi, CBKoX), einem Enstspänroboter (ESR), drei Spannstationen (SPi) zum Aufspannen und Abspannen der zu bearbeitenden Teile auf/von Paletten, Pufferplätzen zum Zwischenlagern und einem alle Einheiten verbindenden schienengebundenen Transportroboter (STR). Wegen der nicht genau positionierbaren tonnenschweren Werkstücke besaß ein Spannplatz ein Mess-Portal zur Ermittlung von Spann-Korrekturwerten.

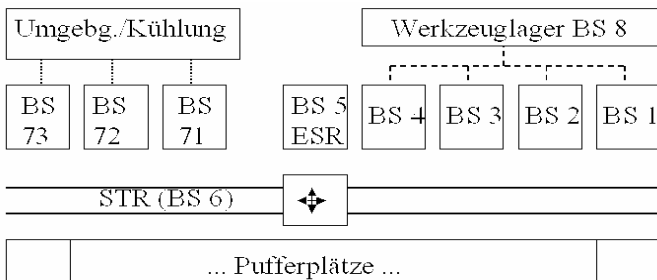


Abbildung 4: Flexibles Fertigungssystem FMS 2500

Die Bearbeitungszentren hatten numerische Steuerungen CNC 600, der Entspänroboter eine Robotersteuerung IRS 713 (eine Weiterentwicklung der IRS 650) und der Schienentransportroboter eine Speicherprogrammierbare Steuerung PC 600. An den Spannplätzen war eine Kommunikation mit den Bedienern über Tasten, Türkontakte und Lampen realisiert. Die Bestückung der Bearbeitungszentren mit den erforderlichen Werkzeugen erfolgte per Hand aus einem Werkzeuglager (LBW, BS 8) über Transportwagen.

3.1 Übersicht über das Steuerungssystem

Die für einen Tag allgemein kapazitätsgeplanten Aufträge wurden dem FMCS übergeben und waren durch das System zu steuern, welches hierarchisch aufgebaut werden sollte.

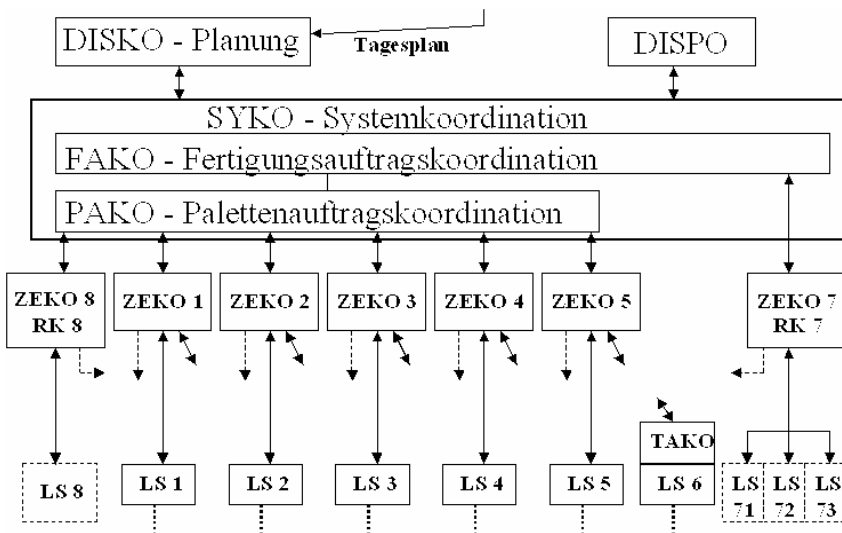


Abbildung 5: Steuerungsstruktur zum FMS 2500

Den lokalen Steuerungen (LS) von Bearbeitungszentren (BS 1-4), ESR (BS 5) und STR (BS 6) wurden also im Zuge einer Prozessualisierung Koordinationssteuerungen übergeordnet. Ebenso erhielten die drei Spannplätze (BS 7i) und das Werkzeuglager (BS 8) als Handarbeits-Stationen je eine Steuerung für die Bedienerkommunikation und die Ressourcenkoordination an Paletten bzw. Werkzeugen. Bei den automatischen Stationen sollten die Koordinationssteuerungen alle für den Fertigungsdurchlauf eng mit ihnen verbundenen Aufgaben im Sinne einer Zelle steuern. Beim STR (BS 6) bestand dann die Hauptaufgabe

darin, aus allen anfallenden Transportanforderungen den optimalen Auftrag auszuwählen und zu steuern.

Für alle Aufträge mit Paletten wurde darauf eine Paletten-Auftragskoordination PAKO übergeordnet. Eine dieser übergeordnete Fertigungs-Auftragskoordination FAKO koordinierte die PAKO mit Spann- und Kühlaufträgen an die Spannplatz-Koordination. Wegen der engen Beziehungen bezeichneten wir PAKO und FAKO als Teile einer Systemkoordination SYKO. Die Kopplung mit der Tagesplanung erfolgte über eine operative Planung DISKO, die die Aufträge mittels mehrerer Prioritätsalgorithmen zwecks Auswahl durch den Dispatcher auf die konkreten Arbeitsstationen verteilte. Zu diesen Grundfunktionen wurden dann weitere Funktionen ergänzt wie z. B. Diagnostik- und Eingriffsfunktionen für den Dispatcher in einem Modul DISPO und Anzeige-, Eingriffs- und Archivierungsfunktionen in den Zellensteuerungen.

Während beim System FMS 2200 diese ähnliche Prozessstruktur mit zwei Rechnern K 1630 und zwei Bürocomputern BC A 5130 implementiert werden musste, was zu Engpässen und geringerer Übersichtlichkeit führte, gelangte beim FMS 2500 von Mikromat ein lokales Netzwerk mit 10 PC vom Typ EC 1834 zum Einsatz. Dadurch bestand ein geringeres Ausfallrisiko für das Gesamtsystem und eine bessere Übersichtlichkeit für alle Bediener. Auch das Testen der Software gestaltete sich durch Implementierung von Prototypen und deren Vortesten in simulierter Umgebung einfacher.

3.2 Virtuelle Zellen

Eine Zellenkoordinationssteuerung ZEKO soll die Abläufe von eng zusammenhängenden Funktionen aufeinander abstimmen, um so die Effektivität deutlich zu erhöhen und die Auswirkungen von Störungen lokal zu halten.

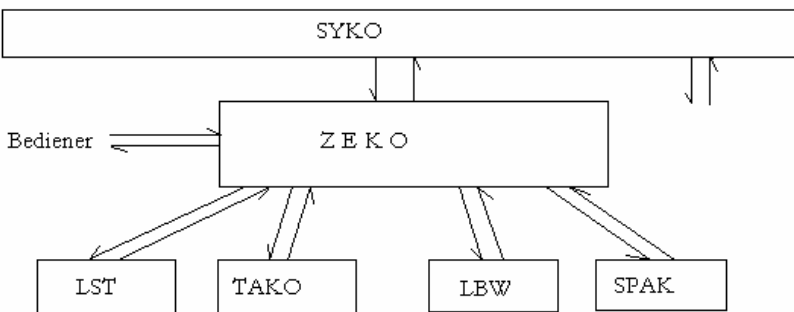


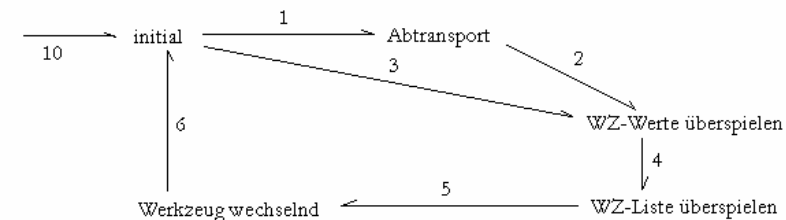
Abbildung 6: Einordnung der ZEKO in das Gesamtsystem

Die ZEKO arbeiteten als Sonderfall jeweils nur mit einer lokalen Maschinensteuerung, betrachteten aber Werkzeugbereitstellung einschließlich Korrekturwerten LBW, Spannplatz-Koordination SPAK und Transport-Koordination TAKO für den STR virtuell untergeordnet als Dienstleister. Die Anforderungen der verschiedenen Zellen mussten dort koordiniert werden. So wurde z.B. die physisch singuläre Ressource Transportroboter zeitweise je nach Bedarf über die TAKO einer der Zellen zugeordnet.

Folgende Funktionsklassen waren zu realisieren:

1. Hauptfunktionen für die Werkstückbearbeitung, abgeleitet von der Basis,
2. Anzeigefunktionen für Arbeitsstand und Ergebnisse,
3. Eingriffsfunktionen zur Fehlerbeseitigung,
4. Simulation fehlender Eingangsinformationen,
5. Servicefunktionen, z.B. Laden/Archivieren von Programmen und Listen und NC-Editorfunktionen,
6. Wiederanlauffunktionen.

Zur Realisierung dieser Aufgaben wurde die ZEKO weiter dekomponiert in eine Zellenauftragskoordination und Prozesse zur Steuerung der lokalen Maschinensteuerung LST. In einer Entitätsklasse „Auftrag“ sind der Einrichteauftrag und der Arbeitsgang zusammengefasst. Die Entitätsklasse „Operation“ steht für die Unterprozesse. Die Entitätsklasse „Operator“ steht für die exklusiv von der ZEKO genutzte Maschine mit lokaler Steuerung und für die von mehreren Zellensteuerungen genutzten TAKO, LBW und SPAK. Alle Entitäten lassen sich mit Zustandsgraphen beschreiben. Beispielhaft soll der um den Zustand „initial“ ergänzte Zustandsgraf für den Einrichteauftrag angegebene werden.



- mit
- 1 : alte Palette vorhanden / Abtransport anweisen
 - 2 : abtransportiert / WZ-Werte anfordern
 - 3 : keine alte Palette / WZ-Werte anfordern
 - 4 : WZ-Werte überspielt / WZ-Liste laden
 - 5 : Liste überspielt / Werkzeugsatzwechsel an Bediener anweisen
 - 6 : Endemeldung des Bedieners / Initialzustand herstellen
 - 10 : Initialisierung oder Auftrag abbrechen / ggfs. Abbruch

Abbildung 7: Zustandsgraf zum Einrichteauftrag

Die Zustandsgrafendateien wurden für den Zustandsgrafeninterpreter SIRIUS 2.2 [1] aufbereitet. Nach Ablauf der Initialisierungen übernahm der Zustandsgrafen-Interpreter die Programmsteuerung, verwaltete alle aktivierten Entitäten, testete ihre Bedingungen und löste die daraus folgenden Aktionen aus. Dadurch wurde eine Pseudoparallelität auf Einprozessorsystemen unter MS DOS gewährleistet.

3.3 Handbetrieb und Wiederanlauf

Entsprechend dem Motto bei UNION Gera: „Alles soll auch noch per Hand möglich sein.“ mussten in einer pseudographischen Darstellung detaillierte Zustandsanzeigen am ZEKO-Rechner implementiert werden und zusätzlich für den Sofort-Überblick eine farbliche Hinterlegung der Teilfenster (schwarz – Ruhe, grün – Arbeit, rot – Fehler). Hierzu war ein Fenster-Tool selbst entwickelt worden. Über die Tastatur konnten dann z.B. verlorengegangene Meldungen und Anweisungen simuliert, fehlende NC-Bearbeitungs-Programme von Diskette eingelesen oder korrigierte NC-Programme auf Diskette archiviert werden.

Wichtige Zustände wurden in regelmäßigen Abständen gesichert. Im Falle eines Wiederanlaufs nach Havarie erfolgte dann eine Herstellung des vorherigen gesicherten Zustands und ggf. eine Nachführung per Hand, wenn die Bearbeitung weitergelaufen war. Lade-Operationen für NC-Programme und Korrekturwerte waren evtl. zur Sicherheit zu wiederholen.

3.4 Weitere Entwicklung

Nach der Inbetriebnahme am 2. Oktober 1989 war Begleitung der Einfahrphase vereinbart, in der die Stilllegung des Entspannroboters aus Wirtschaftlichkeitsgründen erfolgte. Die Erfüllung einer hohen Anzahl eingesetzter Roboter war 1990 nicht mehr erforderlich.

Im ersten Halbjahr 1990 konnten noch Videoaufnahmen gedreht werden. Der Auftrag ist mit der Währungsunion abgeschlossen worden. Die Firma ist dann zweimal verkauft worden, so dass eine Aussage über den weiteren Betrieb nicht möglich ist.

4 Schlussbemerkungen

Die Überführungsarbeiten in die Praxis in so großem Umfang endeten 1990. Heutzutage existieren Firmen, die solche Automatisierungsaufgaben übernehmen und die erforderliche Gewährleistung erbringen können. Es bestehen 20 Jahre später natürlich andere technische Möglichkeiten, aber man führt auch

Wirtschaftlichkeitsbetrachtungen zum Grad der Automatisierung durch, die früher nach erteilten Planvorgaben nicht mehr von großer Bedeutung waren.

Mein Dank für die Unterstützung gilt insbesondere dem heutigen Privatdozenten an der TU Dresden Dr.-Ing. habil. Martin Engeliën. Prof. Stahn ist leider früh im Alter von 64 Jahren verstorben. Von drei jüngeren Mitarbeitern wurden Firmen gegründet.

5 Literatur und Internetquellen

- [1] DOETZKIES, U. (1990): Basis, Steuerung und kooperierende Zustandsgraphen. *Dissertation Technische Universität Dresden, Fakultät Informatik, 100 S.*
- [2] ENGELIËN, M. (1981): ARS – Allgemeine und Rekursive Strukturierung – eine Basistechnologie der Algorithmenprojektierung. *Technische Universität Dresden – Sektion Informationsverarbeitung, Sektionsschrift.*
- [3] ENGELIËN, M. & STAHN, H. (1989): Softwaretechnologie CAMARS-Technologie. *Akademie-Verlag Berlin.*
- [4] ENGELIËN, M. & STAHN, H. (1991): CAMARS-FMCS – Flexible, Distributed Control in Machining Industry. *Systems Analysis, Modelling, Simulation 8, S. 635-643, Akademie-Verlag Berlin.*
- [5] SCHMIDT, E. & STAHN, H. (1984): Programmiersprache und Bedienung der Industrierobotersteuerung IRS 650. *Wissenschaftliche Zeitschrift der Technischen Universität Dresden (4), S. 207-212.*
- [6] SCHMIDT, E. (1991): Zellenkoordinationssteuerung in CAMARS-Technologie. *Im Rahmen des Workshop CAMARS-Steuerungssysteme am 5./6. Dezember 1991 Dresden, 17 S.*
- [7] STAHN, H. & ENGELIËN, M. (1991): CAMARS-FMCS – FMS-Steuerung verteilter Intelligenz, *Fertigungstechnik und Betrieb 41, S. 268-272.*
- [8] STAHN, H. et al. (1989): Flexibles Produktionssteuerungssystem FMCS Variatormontage. *Wissenschaftliche Beiträge zur Informatik – Informatik-Zentrum des Hochschulwesens, Heft 4, S. 27-40.*