

KfK 2766B
Februar 1979

Tuning des Job Entry Subsystems 3 im Rahmen der Einführung des MVS Betriebssystems in der ADI

H. Bachmann, D. Koopmann
Abteilung Datenverarbeitung und Instrumentierung

Kernforschungszentrum Karlsruhe

KERNFORSCHUNGSZENTRUM KARLSRUHE

Abteilung Datenverarbeitung und Instrumentierung

KfK 2766 B

Tuning des Job Entry Subsystems 3 im Rahmen der
Einfuehrung des MVS Betriebssystems in der ADI.

H. Bachmann, D. Koopmann

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe



Zusammenfassung:

Dieser Bericht beschreibt sowohl die Arbeiten zur Anpassung und Leistungsverbesserung des Job Entry Subsystems 3 (JES3) als auch die Entwicklung neuer Systemkomponenten im Hinblick auf die Einfuehrung des Multiple Virtual Systems (MVS) als Grossrechnerbetriebssystem in der ADI.

Tuning of the Job Entry Subsystem 3 in introducing the MVS Operating System at the ADI.

Abstract:

This report describes the task of the tuning and adaption of the Job Entry Subsystem 3 (JES3) as well as the developing of new system components with regard to the introduction of the Multiple Virtual Operating System (MVS) in the ADI.

Inhaltsverzeichnis

	Seite	
1.0	Einfuehrung.	1
1.1	Rechenanlage im ADI.	1
1.2	Das Subsystem JES 3.	2
2.0	Input Stream Manager (ISM) Einfuehrung.	5
2.1	ISM - Input	6
2.2	ISM Jobsteuerung.	6
2.3	Die Job Klassen Zuordnung.	9
2.4	Modul Aufbau und Environment.	15
3.0	Anpassung der Job CPU Zeit und Korrelation der Job Klasse mit der Performance Group des System Resource Managers.	19
3.1	Performance Groups.	19
3.2	Implementierung.	20
4.0	Console Stream Manager (CSM).	23
4.1	Zielsetzung und Vorgabe.	23
4.2	Realisierung und Implementierung.	24
5.0	Anpassung des Jes3 Exits 20.	28
5.1	Der Standard Exit 20.	28
5.2	Die Modifikation.	30
6.0	Anpassung des Jes3 Exits 21.	33
6.1	Der Standard Exit 21.	33
6.2	Die Modifikation des Standard Codes.	33
6.3	Erweiterung des Jes3 Exits 21.	33
6.4	Beschreibung der Druckausgabe.	34
7.0	Displaya: Ein Dynamisches Support Program(DSP) zur Bereitstellung von Job-Status Information fuer den Rechenzentrums-Benutzer.	36
7.1	Aufruf von Displaya.	36
7.2	Erlaeuterungen zur Implementierung.	36
7.3	Eingabeverarbeitung.	37
7.4	Funktionsbeschreibung.	39
7.5	Ermittlung der Job-Status Information.	40
7.6	Ausgabeverarbeitung.	41
7.7	Fehlerbehandlung und Nachrichten.	42
7.8	Environment von Displaya.	42

	Seite
8.0 CLCH: Ein DSP zur Modifikation von Job Klassen durch den Operator.	45
8.1 Aufruf von CLCH.	45
8.2 Erlaeuterungen zur Implementierung.	45
8.3 Eingabeverarbeitung.	46
8.4 Realisierung.	46
8.5 Fehlerbehandlung und Nachrichten.	47
9.0 Literatur-Hinweise.	51
10.0 Abkuerzungen und Begriffe.	52
Jes 3 Control Blocks	53

Vorwort.

Dieser Bericht setzt beim Leser Grundkenntnisse des IBM/MVS Betriebssystem Aufbaus als auch des Jes3 voraus.
Der Leser sollte mit folgenden IBM Broschueren vertraut sein:
GA22-7000, IBM System/370 Principles of Operation.
GC28-0607, Introduction to Jes3.
GC28-0750, VS2/JCL.
GC26-3813, Linkage Editor & Loader.

Im Text benutzte Abkuerzungen und Begriffe werden am Ende des Berichtes unter der Rubrik 'Abkuerzungen und Begriffe', 10.0, angefuehrt.

1.0 Einfuehrung.

Das IBM MVS/JES 3 Betriebssystem ist ein sogenanntes General-Purpose-System, d.h. es soll universell fuer einen moeglichst grossen Kundenkreis verwendbar sein.

Es ist daher Sache des Kunden, das System so zusammenzubauen, wie es sein Rechenbetrieb erfordert.

Bei diesem Zusammenbau gibt es nun wiederum mehrere Stufen und Abschnitte der Abstimmung. Dieser Bericht soll sich auf das Tuning und die Anpassung des Subsystems JES 3 beschraenken. Das Jes3 ist der Nachfolger des Attached-Support-Processor (ASP) Systems, das in der ADI seit Jahren betrieben wurde. Das Subsystem Jes3 ist im Gegensatz zu dem frueheren ASP voll in das MVS integriert und wird als Primary Subsystem bezeichnet.

Das Subsystem selbst besteht aus ca. 340 Programm-Moduln mit ca. 128.000 Zeilen Programm-Code plus ca. 16.800 Makro-Code.

Die Arbeiten, auf denen sich dieser Bericht bezieht, umfassen ca. 25 Module mit ca. 15.000 Zeilen Programm-Code.

1.1 Rechenanlage im ADI.

In dem Rechenzentrum der ADI werden 2 lose gekoppelte (loosely coupled) Rechner unter dem Betriebssystem MVS/JES 3 betrieben. Die Rechner sind IBM Maschinen der Modelle 168 mit 4 MB und 158 mit 3 MB realen Speicher. Bei sogenannten loosely coupled systems (Mehrrechnersysteme) handelt es sich um einzelne Rechner, die entkoppelt, auch vollkommen autonom rechnen bzw. betrieben werden koennen. Die Kopplung wird hardwareseitig durch einen Kanal - Kanal Adapter (CTC) realisiert. Die softwaremaessige Voraussetzung der Kopplung oder eines Rechnerverbundes ist das Subsystem Jes 3, welches unter MVS betrieben wird. Es unterstuetzt die Ankopplung von 8 MVS Rechnern. Zusaetzlich koennen aeltere MVT-Rechner (ASP) angehaengt werden. Insgesamt wird ein kombinierter Verbund von bis zu 32 Rechnern unterstuetzt.

Der Hauptgrund fuer den Betrieb eines solchen Systems ist die effizientere Bewaeltigung des grossen Arbeitsanfalls gekoppelt mit einer besseren Ausnutzung der gegebenen Ressourcen als auch die groessere Transparenz und die damit verbundene leichtere Beherrschung und Steuerung des Rechenbetriebes (operating). Untersuchungen haben gezeigt, dass bei einem Ueberschreiten eines taeglichen Jobdurchsatzes von ca. 300 Jobs ein Rechnerverbund in der angedeuteten Form wirkungsvolleren Betrieb ermöglicht /1/, /2/. Der durchschnittlich-taegliche Jobdurchsatz in der ADI betraegt augenblicklich ca. 800 Auftraege (Jobs) pro Tag (inklusive Wochenende).

1.2 Das Subsystem Jes 3.

Das Jes3 ermöglicht eine zentralisierte Verarbeitung und arbeitet nach dem Master - Slave Prinzip. Wobei der Master (Leitrechner) im Jes3 Terminologie GLOBAL und der Slave als LOCAL bezeichnet wird. Der jeweilige Global ist austauschbar, d.h. jeder MVS Local(Slave) kann bei einem Ausfall des Globals(Master) selbst Global werden. Den prinzipiellen Aufbau stellt Abb. 1 dar.

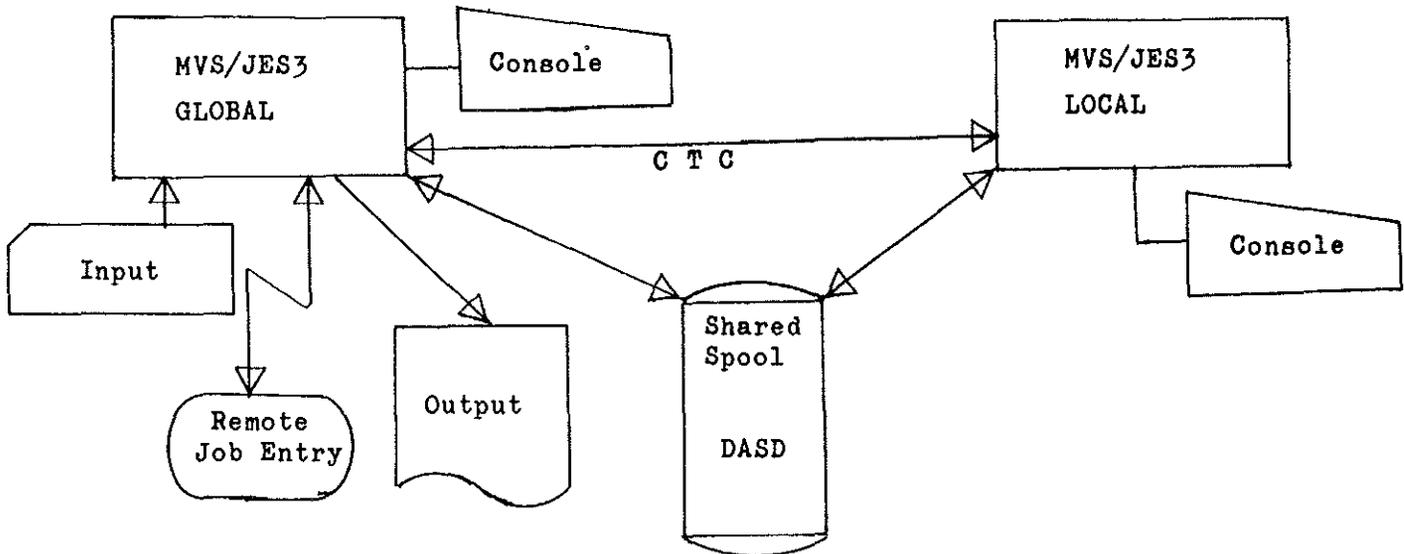


Abb. 1

Die Bezeichnung Job Entry Subsystem ist dahingehend zu interpretieren, als das Jes3 alle anfallenden Vorbereitungsarbeiten als auch die Nacharbeiten fuer den Job abwickelt. Nur das Rechnen selbst (Execution) laeuft unter reiner MVS Kontrolle. Prinzipiell kann man also die Jobverarbeitung in 3 Phasen unterteilen: Vorbereitung, Ausfuehrung, Nachbearbeitung (Abb.2). Die Job-Vorbereitung des Jes3 hat einzig und allein den Zweck dem Prozessor einen Job zur Verfuegung zu stellen, der bereit zur Ausfuehrung ist. Bereit zur Ausfuehrung ist der Job wenn er saemtliche Hilfsquellen(Resourcen) bekommen hat, die er fuer die Ausfuehrung benoetigt. Das koennen Magnetbaender, Arbeitsspeicher, Platten, Lochstreifen usw. sein /3/.

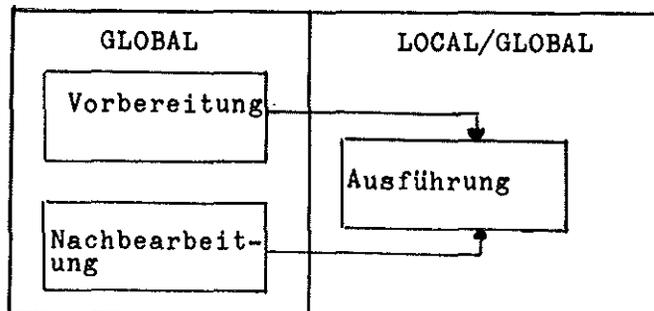


Abb. 2

Da nun verstaendlicherweise die zu vergebenden Ressourcen limitiert sind, muessen dieselben verteilt werden. Diese Verteilung (Scheduling) uebernimmt u.a. Jes 3. Innerhalb des Jes 3 ist der Arbeitsablauf in funktionelle Abschnitte (Segments) aufgeteilt. Sie sind in Abb. 3 vereinfacht dargestellt.

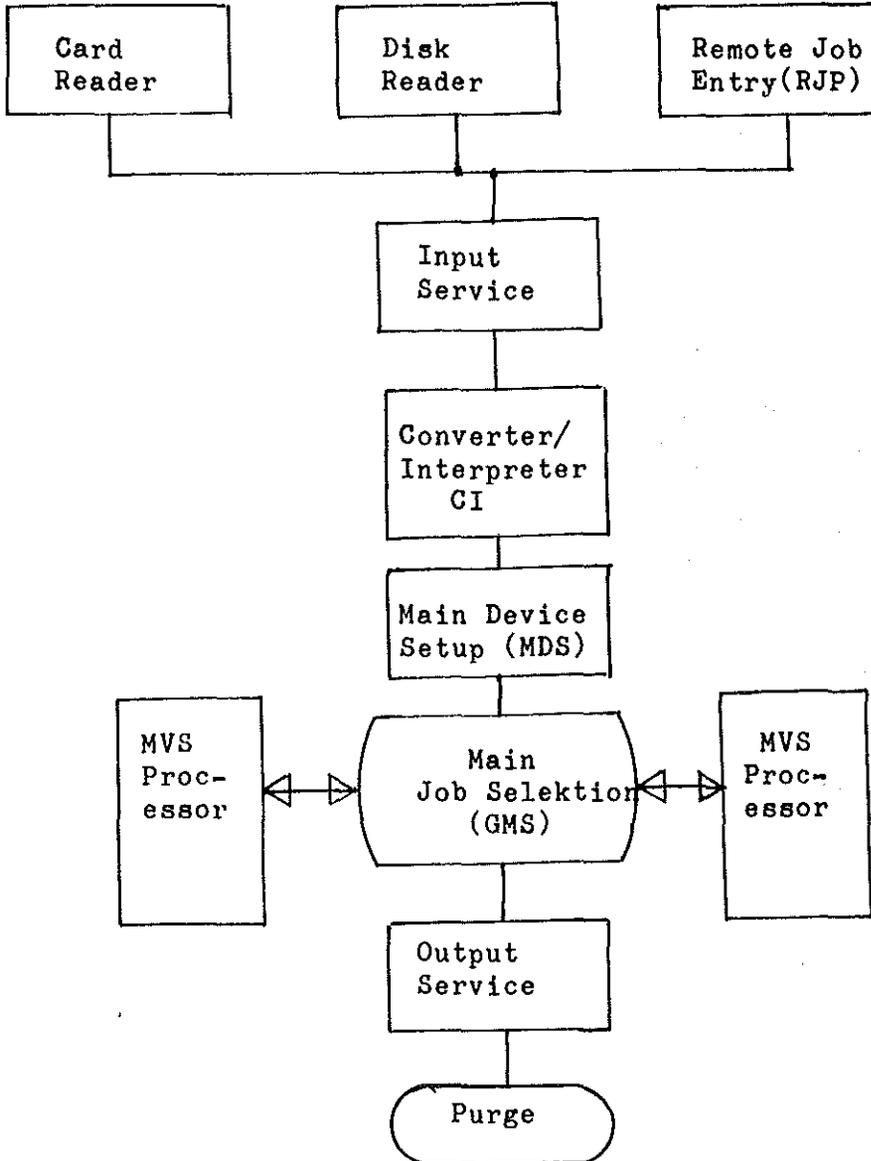
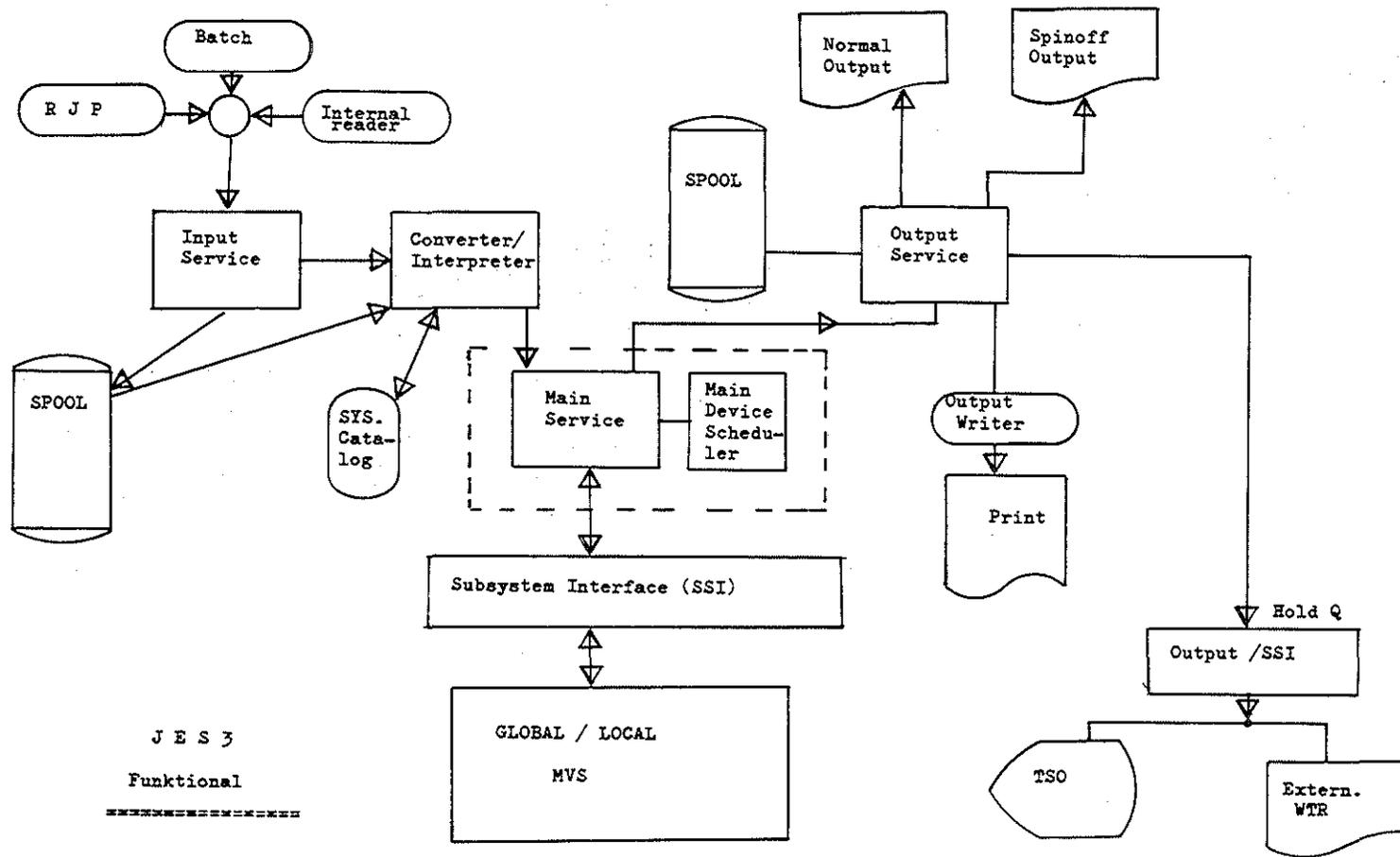


Abb. 3

Diese Segmente stellen gleichzeitig die residenten Standard Dynamic Support Programs (DSP) dar. Ein DSP kann ein oder mehrere Programm-Module enthalten. So enthaelt z.B. der Converter/Interpreter 22 Module. Ein DSP (oder Segment) wird ueber einen dem TCB im MVT analogen Kontrollblock, dem FCT (Function Control Table), gesteuert /4/. Einen funktionellen Ueberblick des JES3 vermittelt Abb.3a.



J E S 3
 Funktional

Abb.3a

2.0 Input Stream Manager(ISM) Einfuehrung.

Der Verteilungsalgorithmus des Jes 3 erlaubt und sieht die Moeglichkeit einer Klassifizierung von Jobs (Job Classes) vor.

Ihr Zweck ist die Zusammenfassung von bestimmten Jobs mit bestimmten Merkmalen /5/.

Die Job-Klasse sagt also etwas ueber die Anforderungen von Ressourcen an den Rechner aus und ist somit ein geeignetes Instrument zur Jobsteuerung, die aufgrund der limitierten und daher zu verteilenden Hilfsquellen erforderlich ist.

In der Praxis wuerde dies dann etwa so aussehen:

Das Rechenzentrum definiert die verschiedenen Klassenmerkmale, z.B. Langlaeufer, viele Magnetbaender usw. Der Benutzer der Anlage muesste jetzt anfangen:

- a) sich seine ganzen Anforderungen zu ueberlegen.
- b) sie auf die Merkmale anwenden.
- c) und schliesslich seinem Job die richtige Klasse geben.

Da dies oft den Benutzer zeitlich oder sonstwie ueberfordert, haben viele Rechenzentren eine Arbeitsvorbereitung eingefuehrt.

Diese Leute sind dann JCL Experten, die diese Aufgabe fuer den Benutzer erledigen.

Fuer die ADI kamen beide genannten Moeglichkeiten nicht in Frage.

Die Arbeitsvorbereitung wuerde sowieso an den engen Stellenplan scheitern, da sie zusaetzliche Manpower erfordern wuerde.

Die benutzerseitige Job-Einordnung in Job-Klassen erscheint aus vielen Gruenden problematisch.

Daher entschied sich die ADI zur Entwicklung eines Algorithmus, dem ISM, der folgende Forderungen zu erfuellen hat:

- a) Benutzerfreundlich, d.h. keinen zusaetzlichen Aufwand von Seiten der Benutzer.
- b) Flexibel bzw. Aenderungsfreundlich bei Feststellung neuer Gegebenheiten.
- c) Leicht wartbar bzw. moeglichst unabhaengig von irgendwelchen Aktivitaeten des Rechnerherstellers.
- d) Jeder Job muss eindeutig definierbar sein.
- e) Operatorfreundlich, d.h. ueberschaubar und leicht beherrschbar zwecks Betriebssteuerung.
- f) Tag- Nachtumschaltung des Rechenbetriebes.
- g) Beruecksichtigung von Spezial Job-Klassen (z.B. 2250) aufgrund ihrer speziellen Anforderungen.

2.1 ISM - Input.

Logischerweise muessen die Input-Parameter fuer den ISM diejenigen Ressourcen sein, die verteilungskritisch sind. Dies sind augenblicklich:

- a) CPU Zeit
- b) Arbeitsspeicher (Region).
- c) Montagearbeiten (Setups) fuer Platten, Baender, Druckerketten, Lochstreifen, spezielle Druckformate, etc, die echten Operatingaufwand erfordern. (real setups)
- d) Druckanforderungen in Zeilen.
- e) Spezial Klassen.

Diese Werte bzw. Angaben sind in der normalen JCL enthalten, die das MVS sowieso zur Job Ausfuehrung benoetigt und daher vom Benutzer sowieso bereitgestellt werden muessen.

2.2 ISM Jobsteuerung.

Ziel und Zweck des ISM's ist es, aufgrund der Input-Parameter den Job X einer Klasse Y mit der Prioritaet Z zuzuordnen. Grundlage der Zuordnung ist eine Job Selektion Matrix (Abb.4), die jeweils den augenblicklichen Gegebenheiten der Installation angepasst ist.

Sollte sich z.B. ein grosser Druckerengpass (zu wenig Drucker) ergeben, so ist der Druckanforderung hoehere Wertigkeit zu geben und Benutzer mit viel Druckausgabe event. in eine andere Klasse mit niedrigerer Prioritaet zu setzen, die tagsueber ueberhaupt nicht mehr bedient wird.

Grundsatzlich gilt fuer die ADI, dass in der Tagschicht Jobs mit kurzer Turn-Around-Zeit bevorzugt werden (Test Jobs).

Jobs, die grosse Ressourcen Anforderungen stellen, werden hauptsaechlich nachts gerechnet. Die Anforderungen werden vom ISM erkannt und der Job wird dann nach der Selektionsmatrix in die entsprechende Klasse gestellt bzw. ihr zugeordnet. Der Betrieb steuert dann wiederum, welche Job Klassen gerechnet werden, indem er die entsprechenden Job Klassen ON oder OFF macht.

Das Job Klassen Geruest, d.h. welche Job Klassen das Rechenzentrum zulaesst, werden in einem, einer Parameterliste aehnlichen, sogenannten Inish-Deck vordefiniert (Abb.4a).

Dies ist notwendig, damit das Subsystem sich bereits bei der Initialisierung seine entsprechenden Tabellen aufbauen kann. /5/

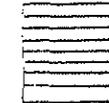
Eine weitere zusaetzliche Moeglichkeit der Jobsteuerung ist, eine ganz bestimmte Klasse X nur waehrend einer bestimmten Tageszeit rechnen zu lassen. Findet der ISM einen solchen Job mit der Klasse X, so setzt er diesen Job automatisch in den Wartestatus (Hold) falls er sich noch ausserhalb des definierten Zeitraumes befindet. Der Operator wird von dieser Aktion durch eine Konsol-Nachricht informiert.

ISM - Job Selektions Matrix

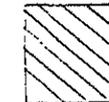
CPU Time	≤ 30 sec		> 30sec - 3min		> 3min - 10min		> 10min - 120min		> 120min		SET- UP
	Lines < 10	Lines > 10	Lines < 10	Lines > 10	Lines < 10	Lines > 10	Lines < 10	Lines > 10	Lines < 10	Lines > 10	
< 1 MB	H(9)	C(7)	B(8)	D(7)	E(6)	I(4)	V(3)	W(2)	Z(1)	Z(1)	= 0
> 1 MB	Q(5)	P(4)	Q(5)	R(4)	T(4)	U(3)	X(2)	Y(2)	Z(1)	Z(1)	
< 1 MB	A(8)	C(7)	J(5)	K(6)	L(5)	N(6)	V(3)	W(2)	Z(1)	Z(1)	≤ 2
> 1 MB	Q(5)	P(4)	Q(5)	R(4)	T(4)	U(3)	X(2)	Y(2)	Z(1)	Z(1)	
< 1 MB	0(4)	1(3)	0(4)	1(3)	0(4)	1(3)	2(2)	2(2)	5(1)	5(1)	3-4
> 1 MB	3(2)	3(2)	3(2)	3(2)	3(2)	4(2)	4(2)	5(1)	5(1)	5(1)	
< 1 MB	6(2)	6(2)	6(2)	6(2)	6(2)	6(2)	7(2)	7(2)	9(1)	9(1)	> 4
> 1 MB	8(1)	8(1)	8(1)	8(1)	8(1)	8(1)	8(1)	8(1)	9(1)	9(1)	

Für die Spezial Klassen
M + G gelten folgende
Limits:
Klasse M (Move+Bibliotheks)
Time(job) = 10min
Region = 256k
Setup = 4
Lines = 20 000

Klasse G (Graphic 2250)
Time(Job) = 30min
Region = 512k
Setup = 4
Lines = 20 000



Jobs werden nur am
Wochenende gerechnet



Jobs werden vorwiegend
nachts gerechnet.



Jobs werden nur bedingt
am Tage gerechnet.

Alle anderen Klassen
werden vorwiegend am
Tage (8:00-18:00) be-
arbeitet.

X(y) X= Job Klassen Name (y) = Job Priorität in der Klasse X

Wenn keine Angaben in der Job Karte bzw. Main Karte gemacht werden
gelten folgende STANDARD (Default) Werte:
Time(Job)= 30sec, Region= 256k, Lines= 5000, Cards= 2000

Abb.4

```
*
*
*           C L A S S E S   &   G R O U P S
*
*   SPECIAL CLASSES
*
CLASS, NAME=F, DEF=YES, GROUP=JGF, IORATE=MED, JOURNAL=NO, X
PRTY=6, SYSTEM=JES3
GROUP, NAME=JGF, EXRESC=(M168,2,,DEMAND,DEMAND), *
EXRESC=(M158,1,,DEMAND,DEMAND)
*
*
CLASS, NAME=S, GROUP=JGS, IORATE=MED, JOURNAL=NO, X
PRTY=10, SYSTEM=JES3, FAILURE=CANCEL
GROUP, NAME=JGS, EXRESC=(M168,1,,DEMAND,DEMAND), X
EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=10
*
CLASS, NAME=G, GROUP=JGG, IORATE=MED, JOURNAL=NO, X
PRTY=10, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGG, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=10
*
CLASS, NAME=M, GROUP=JGM, IORATE=MED, JOURNAL=NO, X
PRTY=7, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGM, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=7
*
*-----*
*   NORMAL CLASSES
*
CLASS, NAME=A, GROUP=JGA, IORATE=MED, JOURNAL=NO, X
PRTY=8, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGA, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=8
*
CLASS, NAME=B, GROUP=JGB, IORATE=MED, JOURNAL=NO, X
PRTY=7, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGB, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=7
*
CLASS, NAME=C, GROUP=JGC, IORATE=MED, JOURNAL=NO, X
PRTY=6, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGC, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=6
*
CLASS, NAME=D, GROUP=JGD, IORATE=MED, JOURNAL=NO, X
PRTY=3, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGD, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=3
*
CLASS, NAME=E, GROUP=JGE, IORATE=MED, JOURNAL=NO, X
PRTY=2, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGE, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=2
*
CLASS, NAME=H, GROUP=JGH, IORATE=MED, JOURNAL=NO, X
PRTY=9, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGH, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=9
*
CLASS, NAME=I, GROUP=JGI, IORATE=MED, JOURNAL=NO, X
PRTY=5, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGI, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=5
*
CLASS, NAME=J, GROUP=JGJ, IORATE=MED, JOURNAL=NO, X
PRTY=4, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGJ, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=4
*
CLASS, NAME=K, GROUP=JGK, IORATE=MED, JOURNAL=NO, X
PRTY=2, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGK, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=2
*
CLASS, NAME=L, GROUP=JGL, IORATE=MED, JOURNAL=NO, X
PRTY=1, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGL, EXRESC=(M158,1,,DEMAND,DEMAND), PRTY=1
*
CLASS, NAME=N, GROUP=JGN, IORATE=MED, JOURNAL=NO, X
PRTY=3, SYSTEM=M158, FAILURE=CANCEL
GROUP, NAME=JGN, EXRESC=(M158,2,,DEMAND,DEMAND), PRTY=3
*
```

2.3 Die Job Klassen Zuordnung.

Eine vereinfachte Darstellung des ISM Arbeitsablaufes ist in Abb.5 dargestellt.

Wie bereits unter 2.3 erwahnt, ist das Ziel des ISM's die Zuordnung des Jobs zu einer Job Klasse + Prioritaet. Ausgangspunkt ist hier die Selektions-Matrix, die in einer Pre-Class-Table umgewandelt wird. Sie ist erforderlich damit eine eindeutige Job/Klassen Zuordnung moeglich ist.

Sie assoziiert die aus den realen Anforderungen des Jobs auf Ressourcen sich ergebenden Pre-Class Namen mit der entsprechenden Job Class/Priority, die dann wiederum in die eigentlichen Jes3 Tabellen neu eingefuegt werden.

So stellt ein generierter Pre- Class Name von z.B. T1R1S1L1 die Anforderung wie $Time \leq 30$ sec CPU Zeit, $Region < 1MB$, $Setup = 0$ (Kein Setup), $Lines < 10$ dar.

Wesentlich ist die Flexibilitaet der Zuordnung via der Class-Attribute-Cross-Reference-Table (Caxtab). Ein Job hat z.B. den Pre-Class Namen T2R2S2L2. Er wuerde in die Klasse R mit der Prioritaet 4 fallen. Umgesetzt mit Hilfe der Selektions-Matrix (Abb.4) sagt z.B. T2 aus, dass der Job eine CPU-Zeit von ≤ 30 sec - 3min anfordert. R2 bedeutet, dass er eine $Region < 1MB$ benoetigt. S2 sagt aus, dass sein (real) Setup-Count ≤ 2 ist und L2 stuft ihn in $Lines < 10$ (10.000) ein. T2R2S2L2 stellt somit eine eindeutige Anforderung dar.

So sollen jetzt z.B. alle Jobs, die diese eine Anforderung haben, auch tagsueber gerechnet werden. Also wird dieses, eine Entry der Pre-Class Table (Abb.6), sagen wir aus der R - Klasse genommen und in die I - Klasse gestellt, die tagsueber online ist. Diese Massnahme stellt natuerlich eine Ausnahme dar, die durch ploetzlich auftretende Engpaesse gerechtfertigt waere. Wuerde es sich um laengerfristige Engpaesse handeln, so wuerde man die Selektions-Matrix aendern.

Die Modifikations-Limit Tabelle (MODTAB) enthaelt sowohl die Input - Parameter Defaults als auch die einzelnen Limits fuer die Anforderungen der Ressourcen, z.B. der CPU-Zeit.

JCT, JQE, RESQ sind reine Jes3 Tabellen, mit deren Hilfe das Jes3 sein Job Scheduling vornimmt. Abb.6a zeigt z.B einen Ausschnitt eines solchen JCT's, der einen fundamentalen job-spezifischen Jes3 Control Block darstellt.

Abb.6b zeigt den JCT in der Kontrollblock-Verkettung/9/. Da zu dem Zeitpunkt, wo dem ISM alle benoetigten Input - Parameter zur Verfuegung stehen, die Jes3-Job-Kette aber bereits vom Jes3 her abgeschlossen ist, d.h. alle Adressen und Pointer gesetzt sind, muss die Kette wiederaufgerissen werden, um die durch den ISM vorgenommene Neuzuordnung in die Kette einzubauen.

Spezial Klassen wie z.B. die grafische Klasse G (2250) werden auf ihre Gueltigkeit und Limits hin ueberprueft. Um Versehen oder Missbrauch zu vermeiden wird eine entsprechende Nachricht an den Operator ausgegeben.

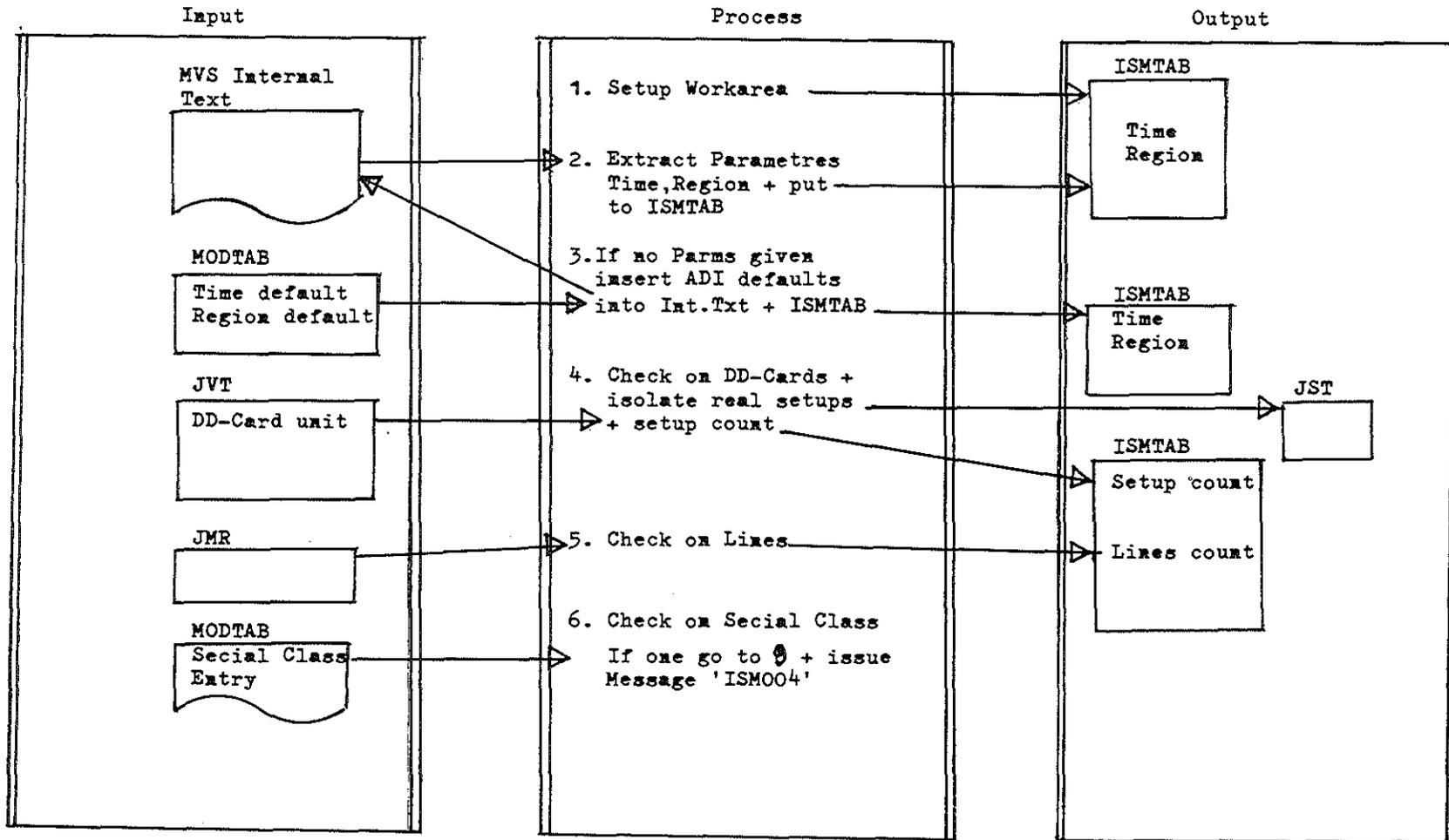


Abb.5(1)

ISM Arbeitsablauf

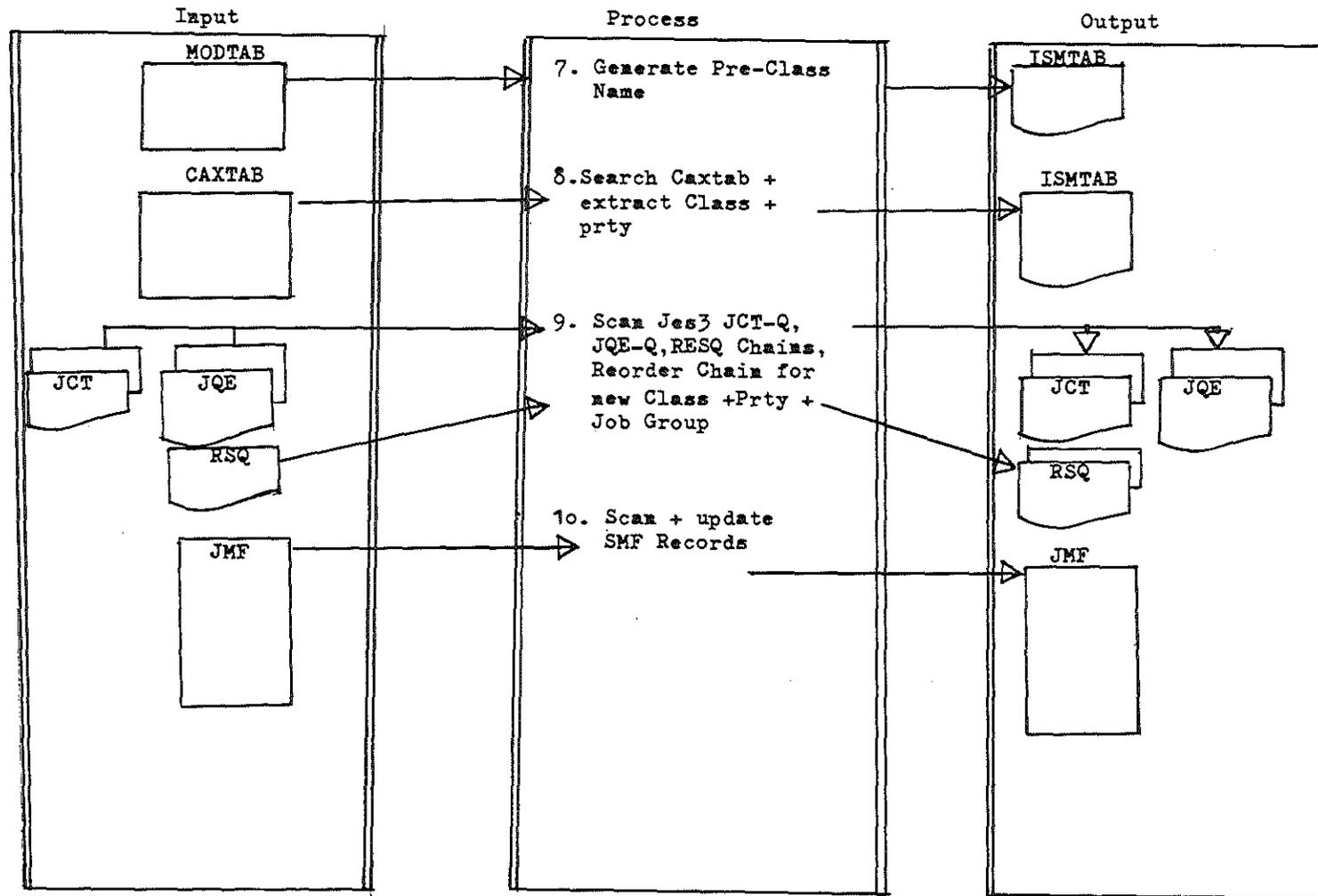


Abb.5(2)

ISM Arbeitsablauf

```

1          CAXTAB TYPE=DSECT
2+CAXTAB  DSECT
3+*****
4+*      CLASS ATTRIBUTE CROSS REFERENCE TABLE      ENTRY      *
5+*                                           *
6+*      PTR CAXTAB : TVTSTAT + 152                    *
7+*****
8+* TABLE PREFIX
9+          DS      OH
10+CAXPRE  DC      AL2(CAXSTRT-*)      PREFIX LENGTH
11+CAXTOTL DC      AL2(ENDCAX-CAXPRE)  TOTAL TABLE LENGTH
12+CAXNO   DC      H'154'      # OF TABLE ENTRIES
13+*-----
14+CAXSTRT DS      OH
15+CAXPCLAS DC     CL9'TIRISIL1'      PRE-CLASS NAME
16+CAXBLK  DC     XL1'40'      FUTURE USE
17+CAXCLASS DC     CL1'D'      JES CLASS NAME
18+CAXCLPTY DC     CL2'08'      JES CLASS PRTY
19+CAXFL1  DC     XL1'00'  FLAG1  SAME AS SPCLFL1
20+CAXFL2  DC     XL1'00'  FLAG2  SAME AS SPCLFL2
21+CAXFUT  DC     CL4' '      FOR FUTURE USE
22+ENDCAX  DS      OX
23          END

```

Abb. 6

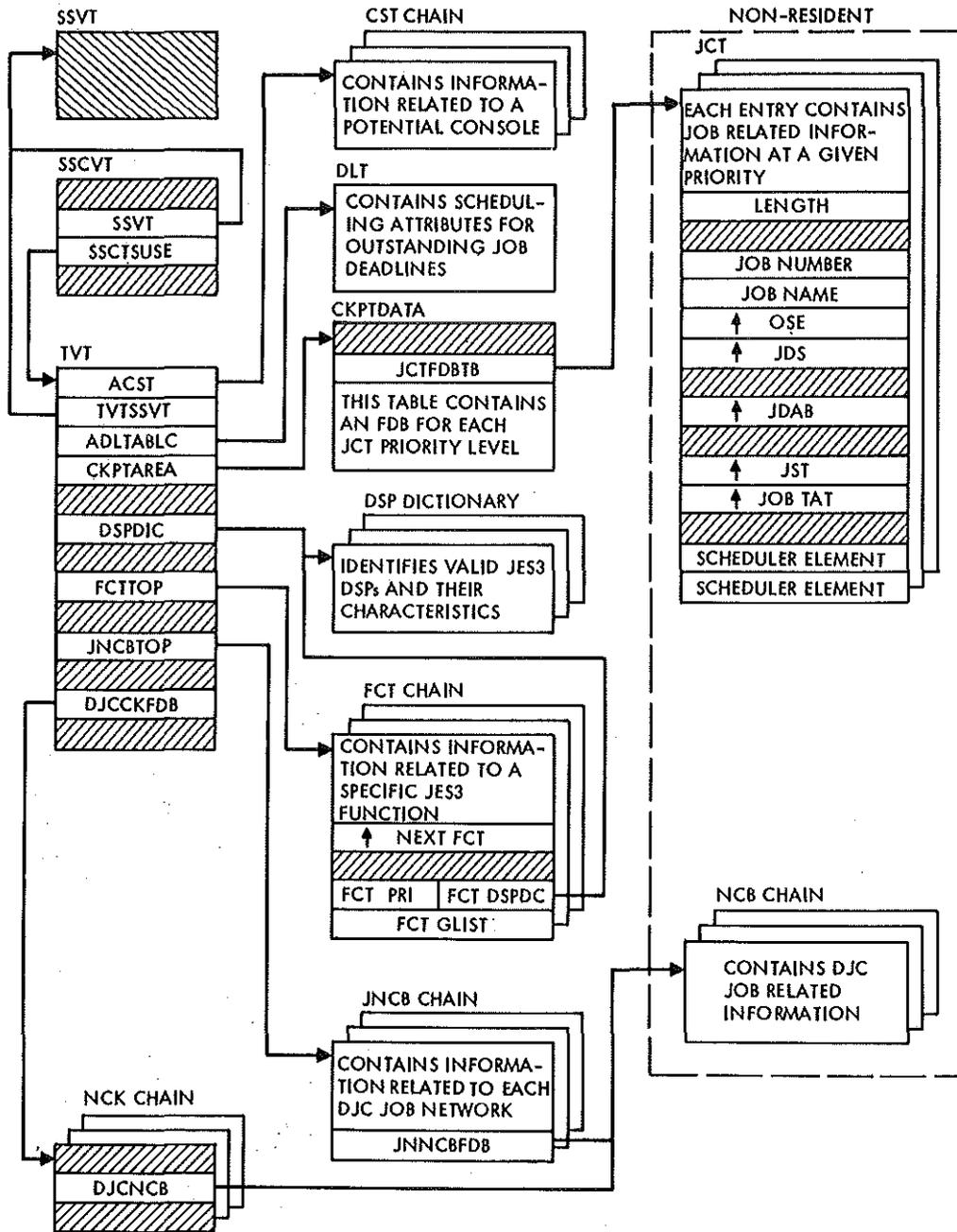
STMT SOURCE STATEMENT

ASM H V 05 16.40

```

3483          IATYJCT
3484+*****
3485+*          FORMAT OF EACH JOB CONTROL TABLE ENTRY *
3486+*****
3487+JCTSTART DSECT
3488+JCTID   DS      F          JCT ID
3489+JCTOTL DS      H -        LENGTH OF ENTIRE JCT - WITH ALL SE'S
3490+JCTFIXL DS      H -        LENGTH OF FIXED PORTION
3491+JCTVARL DS      H -        LENGTH OF EACH SE
3492+JCTJOB  DS      H -        JOB NUMBER (BINARY)
3493+JCTJBNAM DS     CL8 -      JOB NAME
3494+JCTTUSID DS     CL8 -      TERMINAL USER ID
3495+JCTNOSES DS     X -        NUMBER OF SCHEDULER ELEMENTS
3496+JCTNSTEP DS     X -        NO. OF STEPS IN THIS JOB
3497+JCTPRTY DS     X -        JOB PRIORITY
3498+JCTCPTYP DS     X -        TYPE OF CNTRL PGM REQUIRED
3499+JCTLNEST DS     F -        PRINT LINE ESTIMATE
3500+JCTCDEST DS     H -        CARD OUTPUT ESTIMATE
3501+          DS      H          RESERVED FOR JES3          @ZA08518
3502+JCTCDACT DS     H -        ACTUAL CARDS OFF MAIN DIV BY 8
3503+JCTMVSZ  DS     H          MVT REGION SIZE
3504+JCTVS2SZ DS     H          VS2 REGION SIZE
3505+          DS      H          RESERVED FOR JES3
3506+JCTMAINS DS     F -        MAIN INDX FOR THIS JOB
3507+JCTJDFDB DS     XL6        JDAB FDB
3508+          DS      H          RESERVED
3509+JCTFDTAT DS     XL6        JDAB TAT FDB
3510+          DS      H          RESERVED
3511+JCTJDSFD DS     XL6        JDS FDB
3512+          DS      H          RESERVED
3513+JCTOSFFD DS     XL6        FDB FOR OUTSERV ELEMENTS
3514+          DS      H          RESERVED
3515+JCTJMRFD DS     XL6        JMR FDB
3516+          DS      H          RESERVED
3517+JCTSUFDR DS     XL6        JST FDB
3518+JCTUPDTE DS     CL2 -      PRIVATE PROC UPDATE ID
3519+JCTPROC  DS     CL2 -      PRIVATE PROC USED ID
3520+JCTCIERR DS     XL4 -      C/I ERROR CODE
3521+JCTGPRG  DS     CL8 -      DEVICE GROUP ORIGIN
3522+JCTSCHCL DS     CL8 -      SCHEDULER CLASS
3523+JCTDJNET DS     CL8 -      DJC JOB-NET NET-ID
3524+JCTNJPC  DS     CL8 -      NJP CLASS
3525+JCTNJFRM DS     X -        NJP ORIGIN
3526+JCTNJTO  DS     X -        NJP DESTINATION
3527+JCTRSTMN DS     X -        MP SEQ FOR MP JOB WAS ACTIV ON
3528+JCTVSR1 DS     X -        LSTRR FROM CLASS TABLE
3529+JCTSYSTEM DS     X -        SYSTEM TYPE FOR THIS JOB
3530+*-----*
3531+*          DEFINITION OF JCTSYSTEM *
3532+*-----*
3533+JCTSYSJ  EQU     X'04'      ANY JES MAIN
3534+JCTSYSR  EQU     X'02'      ANY REAL MAIN
3535+JCTSYSB  EQU     X'01'      BOTH
3536+*
3537+JCTNJFLG DS     X -        NJP FLAGS
3538+*-----*
3539+*          DEFINITION OF JCTNJFLG *
3540+*-----*
3541+JCTSCHNJ EQU     X'80' -    NJP SCHEDULED BY OPERATOR
3542+JCTNJPRC EQU     X'40' -    NJP SCHEDULED BY PROCESS CARDS

```



JES3 Control Block Chaining

2.4 Modul Aufbau und Environment.

Die in 2.1 erwahnte anstrebare Unabhaengigkeit vom Rechnerhersteller und seinen Wartungsaktivitaeten kann naehrungsweise erreicht werden durch:

a) Benutzung von definierten Exits des Jes3.

b) eigenstaendige DSP's (Dynamic Support Programs).

Naehrungsweise kann sie nur erreicht werden, da die sogenannten System Control Blocks allgemein benutzt werden, d.h. also auch in den Exits.

Der ISM wurde nur mit Hilfe der Exits implementiert. Der Schwerpunkt liegt im Converter/Interpreter DSP (siehe. Abb.3).

Der modulare Aufbau des CI's ist aus Abb.7 zu ersehen /6/.

Der Kern des ISM's (Exit09) liegt am Ende der sogenannten Post-Scan Phase des CI's.

Sieht man sich darauf Abb.8 an, so bekommt man eine Vorstellung, wie der ISM sich in den CI einfuegt, ausgenommen des Exits 3 (IATUX03).

Dieser Exit ist an die CI-Subtask angekoppelt, die wenn sie angefordert wird (posted), als reine MVS Subtask losgeloest vom Jes3 arbeitet. Abb.9 zeigt den modularen Zusammenhang mit dem Exit 3 (IATUX03) innerhalb der Subtask (Der Reader-Interpreter laeuft auf SVS bzw. MVT Maschinen).

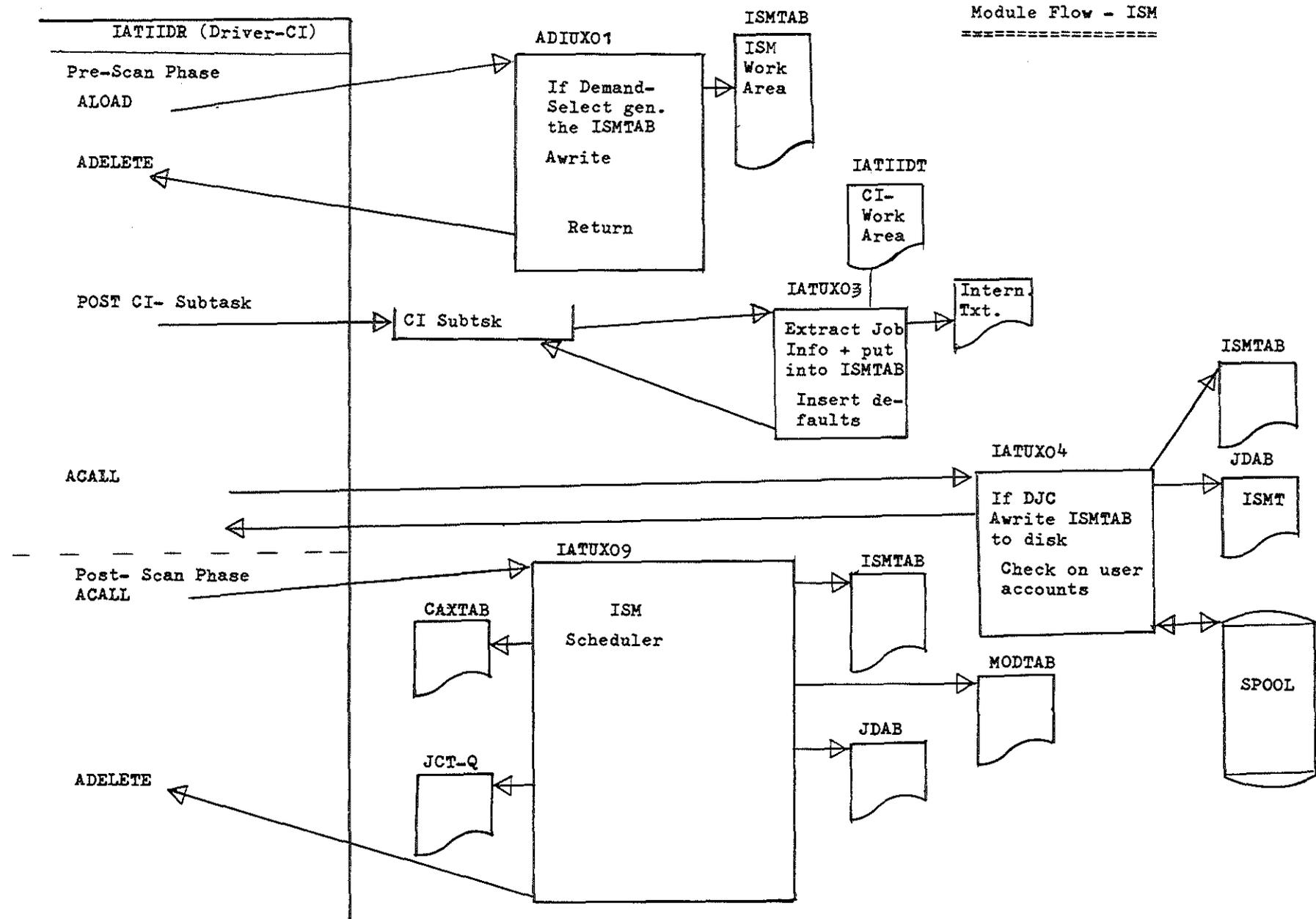


Abb. 8

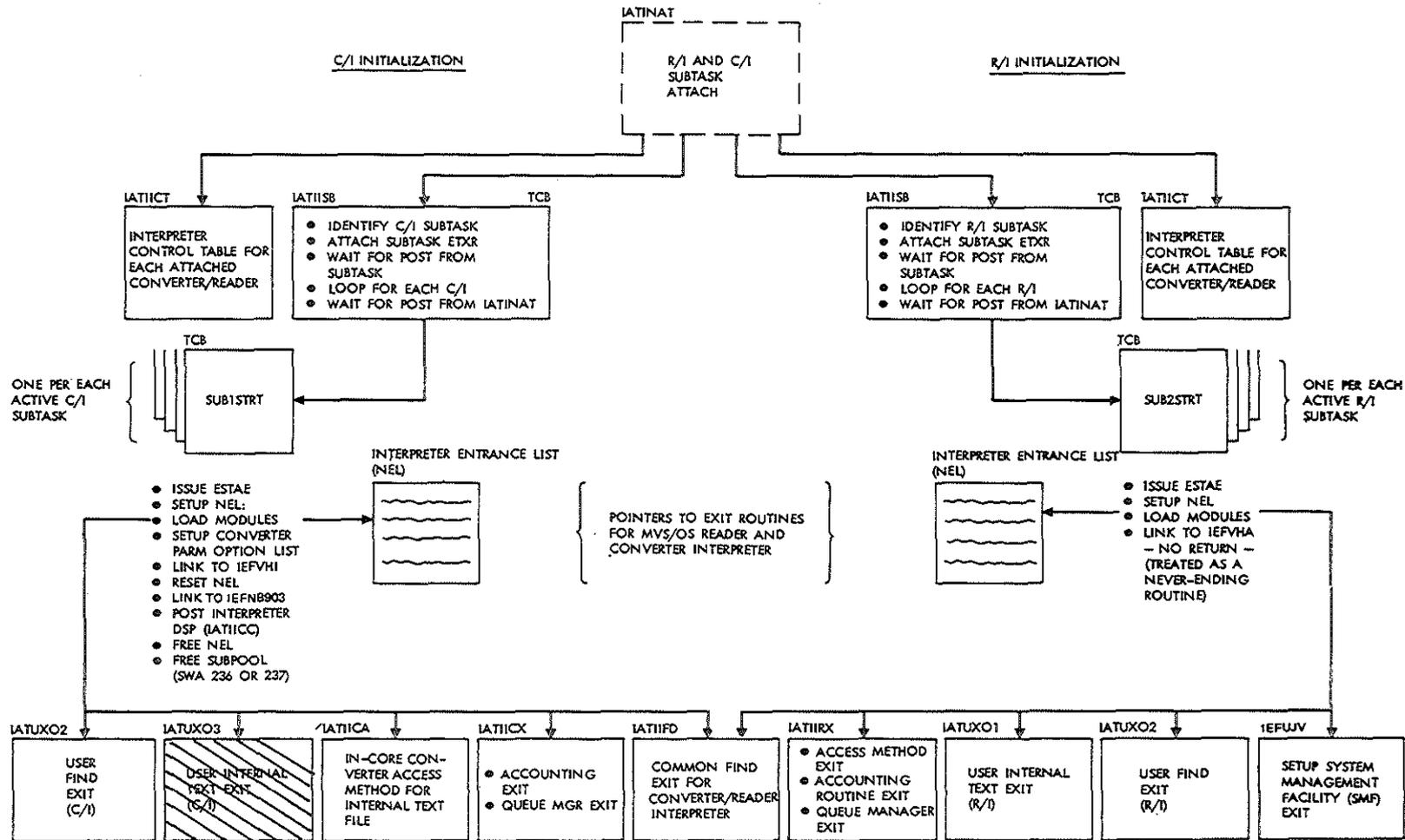


Abb.9

Interpreter Service Control Flow

3.0 Anpassung der Job CPU-Zeit und Korrelation der Job Class mit der Performance Group des System Resource Managers.

Wie bereits erwahnt, betreibt die ADI z.Zt. ein IBM Modell 168 und ein Modell 158 ,die zu einen Rechnerkomplex zu sammengekoppelt sind. (Sieh. 1.1)

Da die Prozessoren eine verschiedene Ausfuehrungszeit bzw. Rechengeschwindigkeit haben,ist eine CPU-Zeit Anpassung fuer Batch Jobs notwendig. So wuerde ein Job auf der 158 ca. 5 mal solange benoetigen als auf der 168. Wird ein Job also auf die 158 geschickt,bekommt er intern automatisch die 5fache von ihm angegebene CPU-Zeit. Somit ist er dem 168-Job CPU-zeitmaessig gleichgestellt.

Die Job Klassifizierung wird, wie in 2.0 angefuehrt,vom ISM vorgenommen. Da der System Resource Manager (SRM) die Job Klassen nicht beruecksichtigt,ist es notwendig die Job Klassen, die nach dem augenblicklich gueltigen Verteilungsalgorithmus vergeben werden, mit dem SRM zu korrelieren,da sonst der SRM,der natuerlich vom ADI Konzept nichts weiss,dieses Konzept ausschalten, zumindest aber einschraenken wuerde. Ausserdem ist die Gefahr des Missbrauches von Seiten der Benutzer gegeben,da die ADI,wie bereits erwahnt,keine Arbeitsvorbereitung betreibt.

3.1 Performance Groups.

Es ist nicht Anliegen dieses Berichtes die Funktionsweise des SRM zu erklaren und wuerde auch weit den Rahmen dieses Berichtes sprengen.Es sei daher nur kurz zwecks Verstaendnis der Modifikation angerissen,was die Performance Group darstellt. Fuer interessierte Leser verweisen wir auf die unter 9.0 angefuehrte Literatur.

Eine Performance Group(PG) wird durch das Rechenzentrum definiert. Sie ist ein Teil der Installation-Performance-System Parameter, die der SRM als Richtlinien und Limits fuer seine Arbeit benoetigt /8/.

Die PG beschreibt die von der Arbeitslast abhaengige Service Rate, die der Job bzw. Step bekommt. Sie stellt eine Zusammensetzung von Performance Charakteristiken dar. In ihr sind z.B. Dinge enthalten wie die Job Domaene,Interval Service Units, Prioritaets Gruppierungen,Performance Perioden Dauer etc. Unter normalen Bedingungen bekommen alle Jobs gute Verarbeitungsraten. Ist die Last auf dem System aber einmal gross, so bekommen bestimmte Performance Gruppen bedeutend niedrigere Raten als andere Gruppen.

3.2 Implementierung.

Fuer die Implementierung wurde ein Jes3 Exit benutzt. Der grobe Arbeitsablauf ist den Abb. 10 zu entnehmen.

Hierzu einige Erlaeuterungen.

Die SWA, die in Abb.10 aufgezeigt ist, stellt die Scheduler-Work Area des MVS dar. Wie bereits in 2.5 erwahnt, operiert der Converter(CI) als Subtask des Jes3. Diese Subtask generiert die SWA und baut in ihr dem MVS verstaendliche Kontrollbloecke auf (Abb. 11). Sie wird dann auf die Spool Datei (DASD) zurueckgeschrieben. Sie ist also gewissermassen eine Kommunikation-Area zwischen Jes3 und MVS. Ausserdem hat sie den generellen Zweck, die Job Queue Zugriffe (I/O) zu reduzieren.

Geht der Job nun in die Ausfuehrungsphase (Execution), so wird die SWA wieder vom Spool eingelesen und durch das 'SWA Create' des Initiators neu aufgebaut.

Der Jes3 Exit laeuft unter dem Task Control Block (TCB), d.h. unter der Kontrolle des Initiators, also im User Address Space. Hierzu kurz einige Erlaeuterungen.

Die Anzahl der Initiator und damit die Anzahl der parallel laufenden Tasks, ist von der Recheninstallation vorgegeben. Sie ist unter anderen abhaengig von der Systemlast. Wird also ein Initiator in einer Job Gruppe, in der sich wiederum die Job Klassen befinden, wieder verfuegbar, so verlangt dieser vom Jes3 die Zuteilung eines Jobs dieser Job Gruppe bzw. der Job Klasse (Job Select). Der Grob Ablauf eines solchen 'Job Select' als auch der modulare Aufbau mit dem Jes3 Exit ist aus Abb. 12 zu entnehmen.

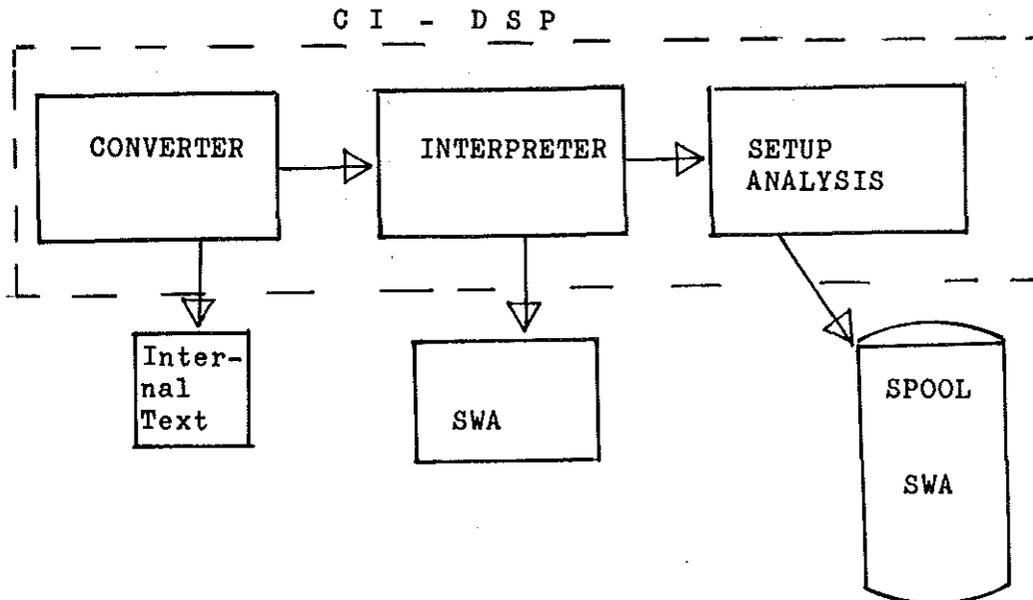


Abb.11

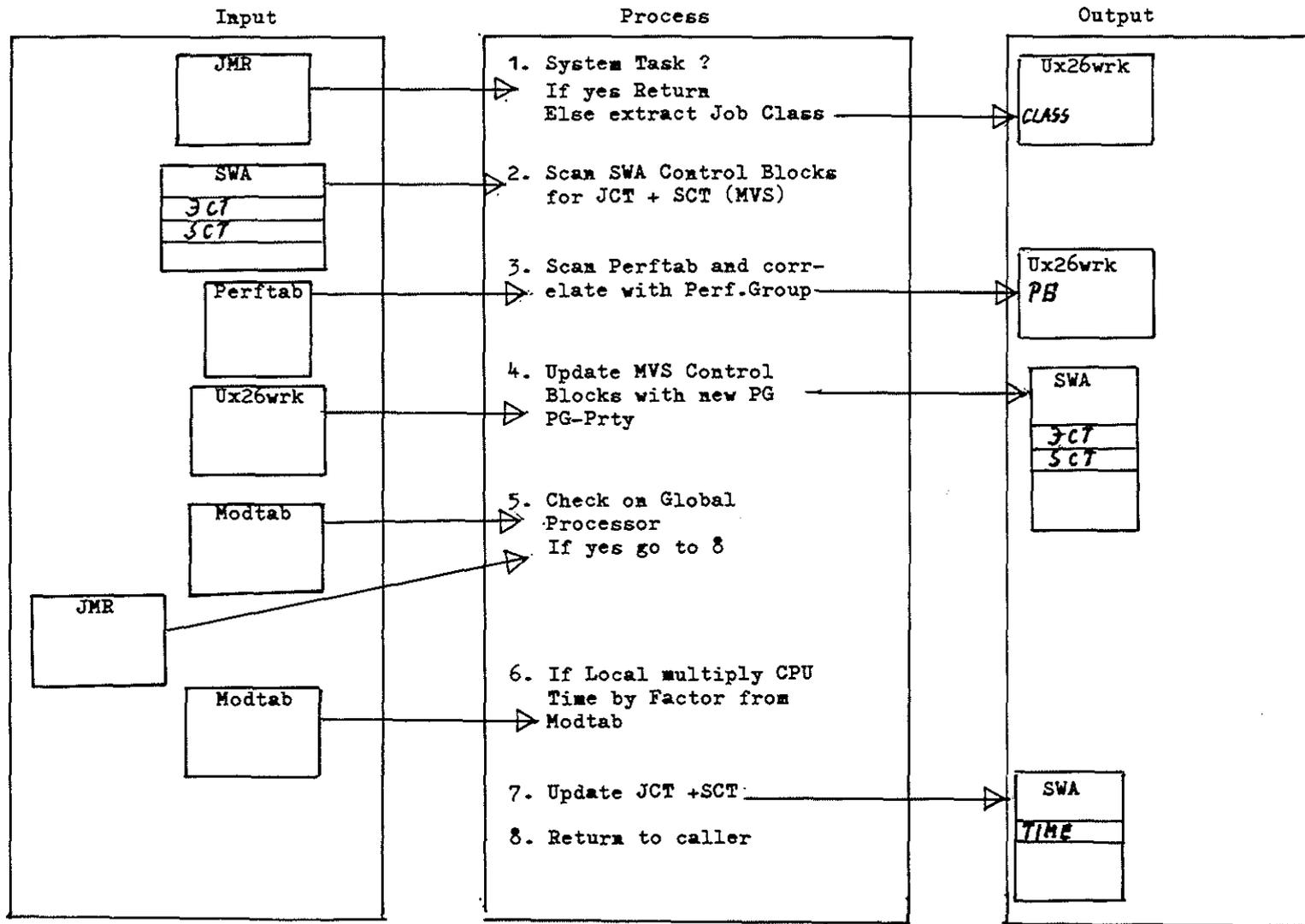


Abb.10

4.0 Console Stream Manager (CSM).

Schon im MVT/ASP System wurde der Operator mit einer Flut von Nachrichten(Messages) und Meldungen des Systems bedacht. Im MVS/JES3 hat diese Informationsflut noch erheblich zugenommen. Im Betrieb stellt sich dann heraus, dass viele der Nachrichten fuer unsere Installation ueberfluessig sind. Wichtige Nachrichten wiederum sind vom System her schlecht aufbereitet, d.h. schwer in der Masse der Nachrichten zu erkennen.

Andere Nachrichten wieder werden an ganz bestimmten Konsolen benoetigt. Das Jes3 bietet zwar hier die Moeglichkeit Nachrichten- Gruppen, sogenannte Funktionsgruppen, auf bestimmte Konsolen zu legen, nicht aber die Moeglichkeit eine bestimmte Nachricht X aus der Gruppe Y auf die Konsole Z zu bringen.

4.1 Zielsetzung und Vorgabe.

Eine gute Nachrichtentransparenz als auch die Reduzierung auf das Wesentliche traegt unmittelbar zum effizienteren Operator Einsatz und damit auch mittelbar zu einer kuerzeren Job-Turn-Around-Zeit bei.

Aus diesen Grund entschloss sich die ADI einen Algorithmus, den CSM, zu entwickeln, der folgende Zielsetzung haben soll:

- a) Unterdruecken von fuer den Betrieb ueberfluessigen Nachrichten.
 - b) Hervorheben bzw. Markieren von wichtigen Nachrichten (z. B. die eine sofortige Aktion des Operators erfordern).
 - c) Dirigieren von spezifizierten Nachrichten an definierte Konsolen.
 - d) Umbau von speziellen Nachrichten in eine andere Form.
- Ausserdem sollen die jeweiligen Nachrichten, die durch den CSM bearbeitet werden, leicht auswechselbar, loeschbar, und Neuzugaenge moeglich sein.

4.2 Realisierung und Implementierung.

Der CSM wurde mit Hilfe eines Jes3 Exits in das Subsystem implementiert. Die zu bearbeitenden Messages sind in einer Tabelle, der Mestabc, abgespeichert. Abb. 13 zeigt einen Eintrag (Entry) dieser Tabelle. Die Tabelle ist im bucket-sort geordnet und enthaelt ISAM-aehnliche Ueberlauf-Bereiche fuer Neu-zugaenge.

Abb. 14 zeigt den Grob- Arbeitsablauf. MVS WTOR's (Write to Operator and Reply) werden grundsaeztlich hervorgehoben. Aenderungen in der Mestabc sind unproblematisch.

Abb. 15 zeigt den Exit mit den CSM im modularen Zusammenhang im Console Service Output-Processing. Device Depend- andt Routines sind Geraete-spezifische Routinen mit ent- sprechend auf das Geraet (z.B. eine IBM 1052 Konsole) abgestimmte Kanalprogramme.

```

1          MESTABL TYPE=DSECT
2+*****
3+*          CONSOLE STREAM MANAGER MESTABL ENTRY *
4+*****
5+MESTABL  DSECT          CSM MESTABLE ENTRY
6+          DS          CF
7+MESNAM   DC          CL4'IAT '          LEVEL KEY
8+* IF LEVEL KEY BLANK, ENTRY AVAILABLE
9+MESFLG1  DC          XL1'CC'
10+LENTY   EQU          X'80'          LAST ENTRY OF TABLE LEVEL
11+LSTLEVL EQU          X'40'          LAST LEVEL
12+FULLOVFL EQU          X'20'          OVERFLOW FULL OF THIS LEVEL
13+FLEVEL  EQU          X'10'          1ST LEVEL ENTRY
14+OVFLAST EQU          X'08'          EOF
15+*
16+MESFLG2 DC          XL1'0C'
17+SUPPRESS EQU          X'80'          DELETE THE MSG
18+EMPHSIZ EQU          X'40'          EMPHASIZE THIS MSG
19+REROUTE EQU          X'20'          REROUTE THIS MSG
20+MLOG    EQU          X'10'          DELTE EXCEPT MLOG
21+DEADMSG EQU          X'08'          DEAD ENTRY
22+*
23+          DC          XL2'0C'
24+*
25+CNSDEST DC          F'0'          CNS NUMBER + CNS DEST
26+LEVELADR DC          A(0)          ADDR OF TABLE LEVEL
27+*-----
28          END

```

Abb.13

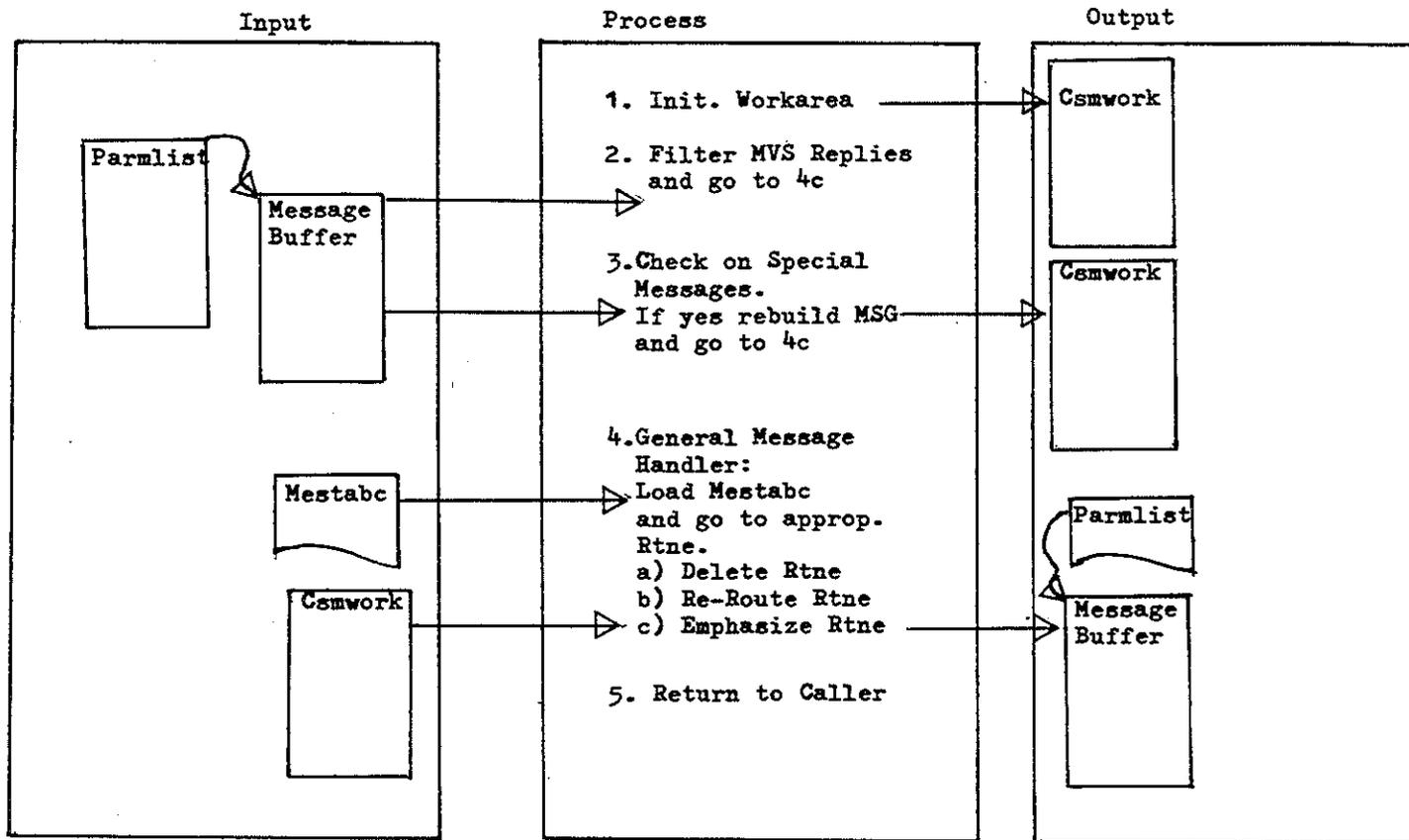
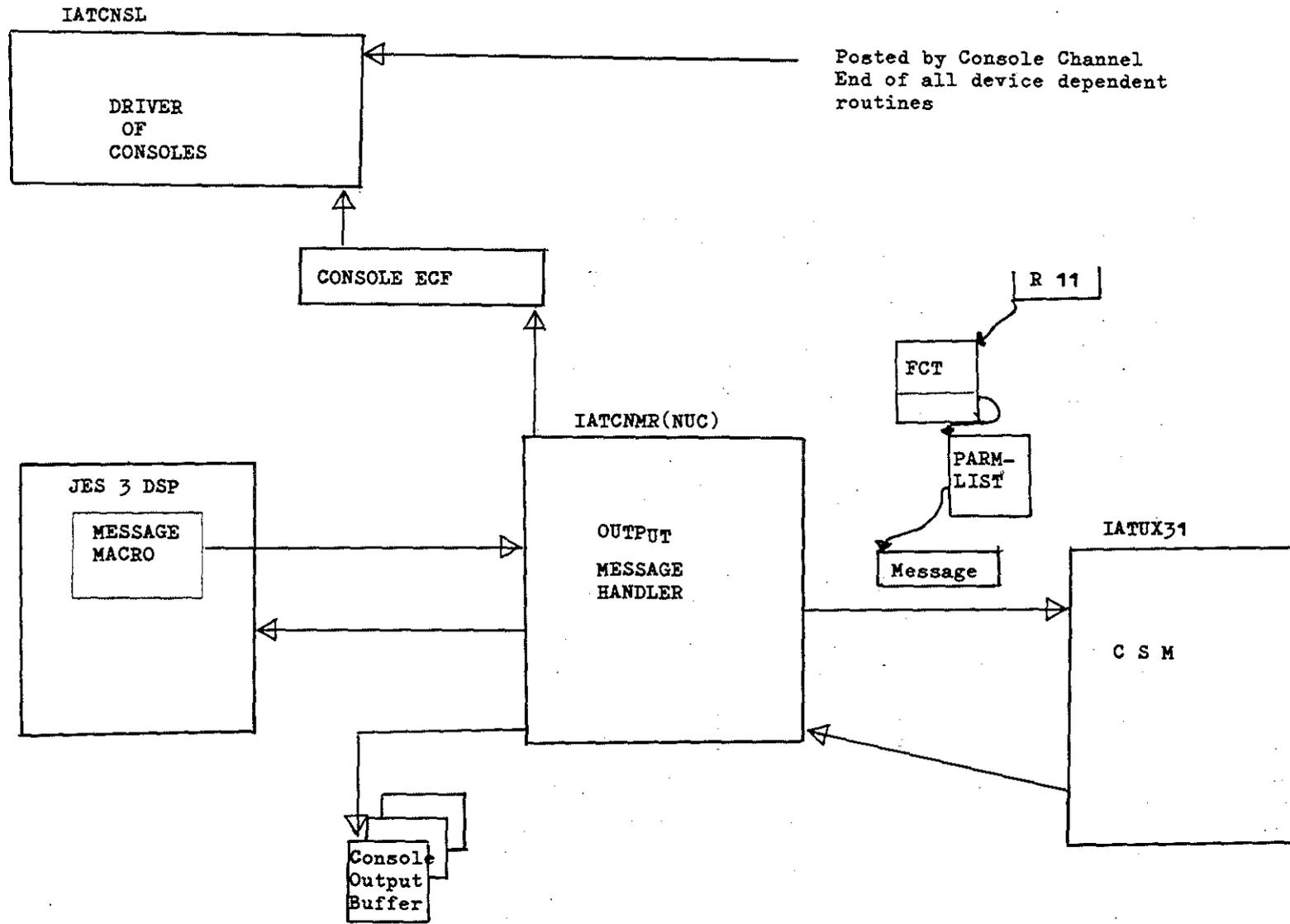


Abb. 14



Posted by Console Channel
End of all device dependent
routines

OUTPUT MESSAGE PROCESSING

Abb. 15

5.0 Anpassung des Jes3 Exits 20.

Die meisten Jes3 Exits sind praktisch 'leere' Exits, d.h. in ihnen ist kein Code enthalten. Es gibt aber einige Exits, die einen sogenannten Standard-Code bereitstellen. Der Benutzer (das Rechenzentrum) kann dann diesen Code implementieren oder ihn durch seinen eigenen ersetzen. Der Exit 20 (IATUX20) stellt Standard Code bereit.

5.1 Der Standard Exit 20.

Dieser Exit des Output Service DSP (sich. Abb.3) generiert aufgrund der Job Information eine sogenannte Kopf- bzw. Titelseite (Header Page) des Jobs fuer den Schnelldrucker. Diese Titelseite ist nicht nur fuer den Benutzer von Beduetung, sondern ganz besonders fuer das Rechenzentrum, da sie die Job Separierung fuer den Operator ermoeeglicht. Waere sie nicht, so wuerden wohl die meisten Jobs nicht in die Haende ihres Besitzers gelangen.

Der Job Name stellt einen Charakter-String dar, der zu einem Tabellen Index uebersetzt wird. Dieser Index wird dann benutzt, um eine Bit-Matrix zu adressieren, mit deren Hilfe dann Zeile fuer Zeile die grossen Blockbuchstaben der Titelseite aufgebaut werden. Die Zeilen werden erst einmal gesammelt und dann direkt an einen Kanalprogramm uebergeben, um sie dann auszudrucken. Abb.16 zeigt so eine Bit- Matrix.

5.2 Modifikation.

Die Information der Original-(nicht modifiziert) Titelseite befriedigte weder die Beduerfnisse der ADI noch die der Benutzer. Sie enthielt wohl den notwendigen Job-Namen, nicht aber die z.B. gewuenschte Job Klasse. Viele Benutzer benoetigten ausserdem zwecks Dokumentation das Datum in leicht lesbarer Form auf der Titelseite. Das Original dagegen enthielt den Namen des Druckers (z.B.PR1) und den Druckertyp (z.B. PRT1403),eine fuer die ADI irrelevante Information. Es war also eine Modifikation des Codes im Exit 20 erforderlich.

Abb. 17 zeigt wiederum den groben Arbeitsablauf.

Der in Abb. 17 unter Punkt 5 angefuehrte Page Counter bewirkt einen dreimaligen Ausstoss der Titelseite. Dies ist notwendig,um dem Operator die Job-Separierung am Drucker zu ermoeeglichen,d.h. das richtige Durchtrennen des Papierstapels im Drucker p. Job.

Das Modul IATOSPR enthaelt die Kanalprogramme,die dann die konvertierten Daten ausdrucken.

Abb. 18 zeigt den Exit 20 im modularen Zusammenhang mit dem Output Service DSP.

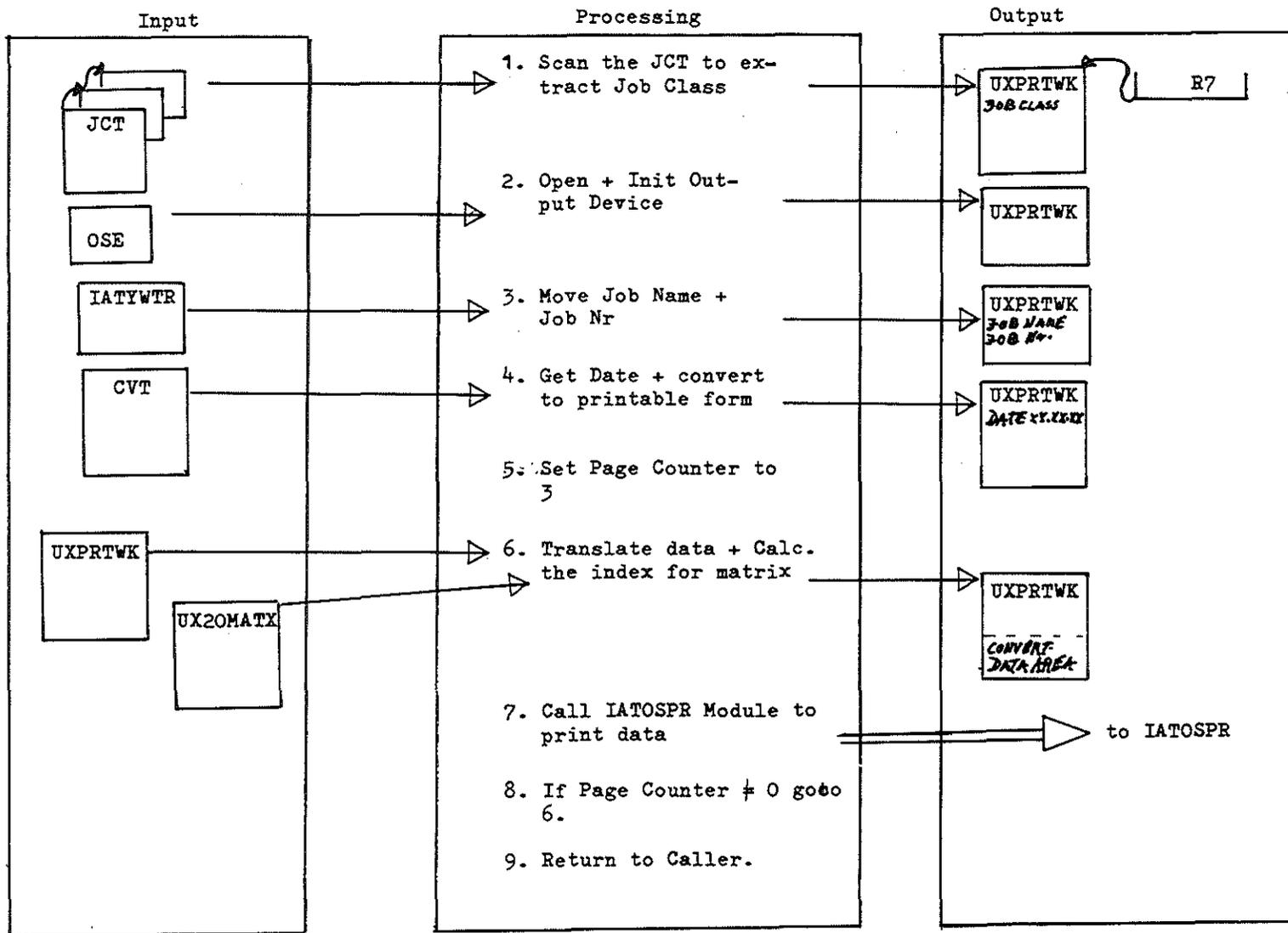


Abb.17

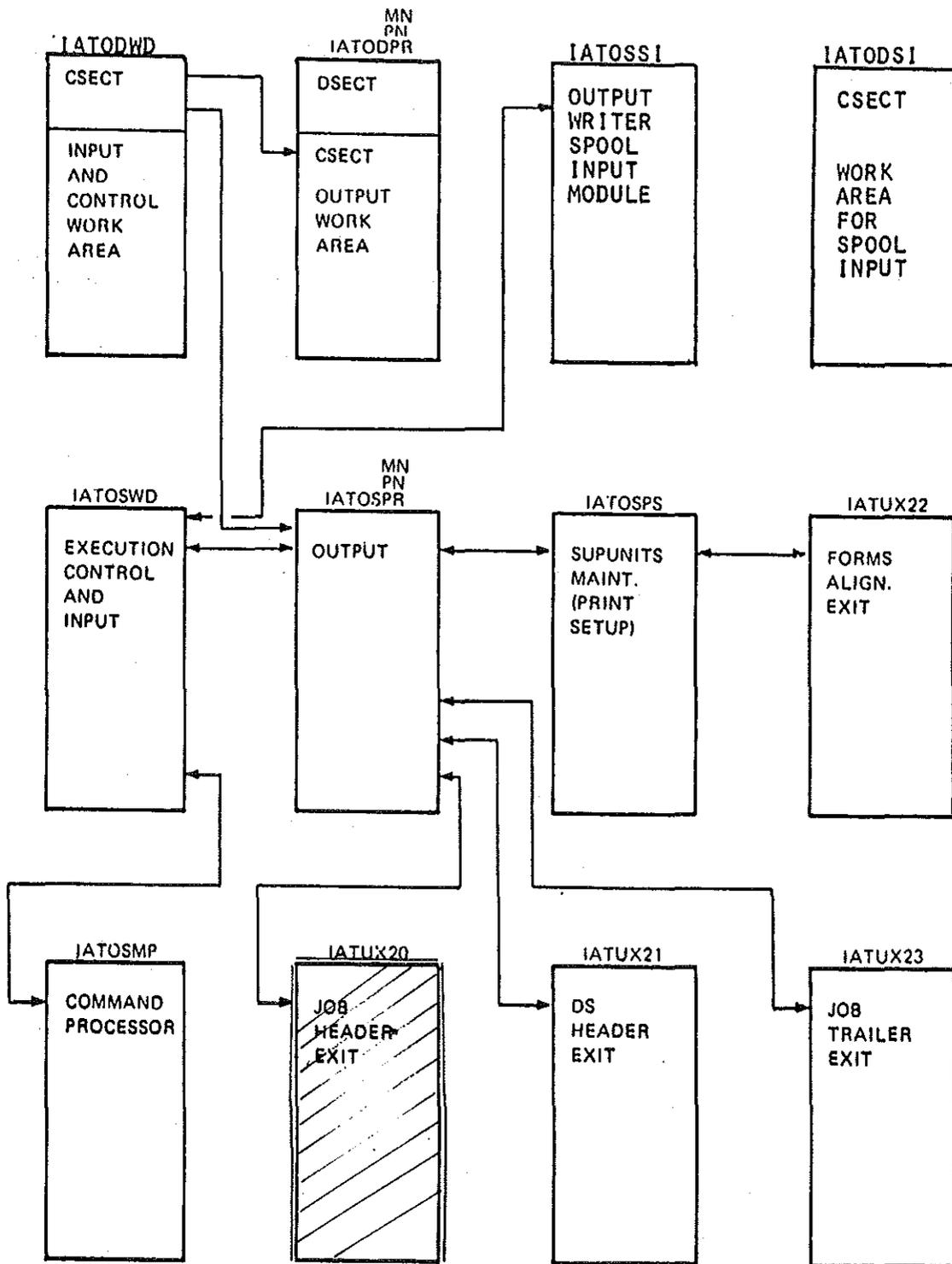


Abb. 18

6.0 Anpassung des JES3 Exits 21

6.1 Standard-Exit 21

Beim Exit 21 handelt es sich, wie unter 5.0 erwähnt, um einen Standard Exit mit bereits implementierten Code. Dieser Exit des Output Service DSP generiert für jede Druckdatei des einzelnen Jobs eine Titelseite (Header Page). Auch hier wird nach demselben Schema der indizierten Bit-Matrix (siehe Abb. 16) verfahren, um die großen Blockbuchstaben zu erzeugen.

6.2 Modifikation des Standardcodes

Der Standard Exit 21 produziert für jede Druckdatei eine Titelseite, also auch für die Dateien JESMSG, JESJCL und SYSMSG, die JES b.z.w. MVS System Nachrichten enthalten.

Der ADI erschien die Produktion dieser drei Titelseiten überflüssig. Außerdem wurde die Übersichtlichkeit für den Benutzer beeinträchtigt, abgesehen von der Papierersparnis von ca. 3000 Seiten pro Tag.

Der Code des Exits wurde dahingehend geändert, daß der Ausdruck der Titelseiten dieser Dateien unterdrückt wird. Der modulare Zusammenhang innerhalb des Output Service DSP ist aus Abb. 18 zu ersehen.

6.3 Erweiterung des JES3-Exits 21

Anstatt der Titelseite für die Datei JESJCL werden in der Ausgabe jedes Benutzerjobs folgende Angaben ausgedruckt:

- a) die Anzahl der eingelesenen Karten
- b) die Anzahl der gedruckten Zeilen für jede Druckdatei
- c) die Gesamtzahl aller gedruckten Zeilen
- d) die Anzahl der gestanzten Karten für jede Stanzdatei
- e) die Gesamtzahl aller gestanzten Karten
- f) das Datum und die Start-/Stop-Zeiten für die Standard-Schedulerelemente

Besonders die Angaben der obigen Punkte b) bis e) können dem Benutzer bei der Abschätzung der LINES- und CARDS-Parameter in der /*MAIN-Karte (siehe /10/) eine gewisse Hilfestellung leisten, falls der Job mit geänderter Eingabe nochmals durchgeführt werden soll.

In der Erweiterung des Exits werden folgende JES3-Kontrollblöcke und -Makros benutzt:

Kontrollblöcke: FRP, JCT, JDAB, JDS, JMR (siehe /9/ und Abb. 6b)
Makros: AGETMAIN, APUTMAIN, ARELEASE, IATXJCT, IATXOSOD, IATXOSOC,
IATXOSOO, IATXRELC (siehe /4/)

Der notwendige Arbeitsspeicher in der Erweiterung wird mittels der Makros AGETMAIN und APUTMAIN am Anfang beschafft und am Ende freigegeben, da der User-Exit 21 als RE-ENTERABLE (Linkage Editor Attribut) (siehe /11/) gekennzeichnet ist.

Die Informationen über die Anzahl der Zeilen, Karten u.s.w. werden folgenden JES3-Kontrollblöcken, die mittels der Makros JESREAD, ARELEASE und IATXRELC aus der Spooldatei geholt bzw. an sie zurückgegeben werden, entnommen:

- a) JDAB(job descriptor block): die Anzahl der eingelesenen Karten
- b) JDS (job dataset control block): Dieser Kontrollblock enthält die Anzahl der gedruckten Zeilen bzw. der gestanzten Karten und die Information, ob es sich um eine Druck- oder eine Stanz-Datei handelt. Zusätzlicher Bestandteil dieses Blocks ist der COPIES-Parameter (siehe /10/) der DD-Karte, der bei der Berechnung der Zeilen-/Karten-Anzahl berücksichtigt wird.
- c) FRP (/*format parameter buffer): Die Angaben auf einer /*FORMAT-Karte über die Anzahl der Kopien (COPIES-Parameter) einer Datei oder, ob eine Druck-/Stanz-Datei zusätzlich gestanzt b.z.w. gedruckt werden soll, sind in diesem Block gespeichert. Diese Informationen werden ebenfalls zur Berechnung der Gesamtanzahl der Zeilen bzw. Karten benutzt.
- d) JMR (job management record): Das Datum und die Start-/Stop Zeiten für die Scheduler-elemente sind in diesem Kontrollblock enthalten. Die folgenden standardmäßigen Scheduler-elemente werden berücksichtigt: Reader(RDR), Converter/Interpreter(CI), Main Service(MAIN), Output Service(OUTS).

6.4 Beschreibung der Druckausgabe

Alle Angaben über die Anzahl der Zeilen/Karten, das Datum und die Start-/Stop-Zeiten, die in den Kontrollblöcken binär dargestellt sind, werden in die alphanumerische Darstellung umgewandelt und in einen Ausgabepuffer gespeichert, der Bestandteil des Arbeitsspeichers ist. Dieser Ausgabepuffer wird mittels der Makros IATXOS00, IATXOS0D und IATXOS0C (siehe /4/) anstatt der Titelseite JESJCL in die Jobausgabe transferiert. Ein Beispiel für die Druckausgabe der Erweiterung ist aus Abb. 19 zu ersehen.

```
*****
*
* TOTAL CARD-INPUT: 00026
*
* DDNAME CLASS LINES
* JESMSG A 000016
* JESJCL A 000040
* SYSMMSG A 000087
* SYSPRINT A 000047
* SYSLOUT A 000005
* FT06F001 A 000511
* FT02F001 A 000203
* FT08F001 B 000700
* TOTAL LINES: 0001609
*
* DDNAME CLASS CARDS
* FT07F001 B 000270
* FT08F001 B 000700
* TOTAL CARDS: 0000970
*
*****

*****
RDR /START/ DATE: 79.024 TIME: 12.34.42
RDR /STOP / DATE: 79.024 TIME: 12.34.43
*****
CI /START/ DATE: 79.024 TIME: 12.34.45
CI /STOP / DATE: 79.024 TIME: 12.34.47
*****
MAIN /START/ DATE: 79.024 TIME: 12.34.47
MAIN /STOP / DATE: 79.024 TIME: 12.34.57
*****
OUTS /START/ DATE: 79.024 TIME: 12.35.12
*****
```

Abb. 19 Beispiel für eine Druckausgabe des Exits 21

7.0 DISPLAYA: Ein Dynamisches Support Program (DSP) zur Bereitstellung von Job-Status Informationen für den Rechenzentrums-Benutzer

DISPLAYA wurde als JES3-DSP konzipiert, um Informationen über den Status eines bestimmten Jobs auf einer Konsole für einen Operator oder über den Status der gesamten Jobqueue auf einem Drucker für die Rechenanlagenbenutzer in übersichtlicher Form zu erhalten. Weitere Funktionen von DISPLAYA werden in Kapitel 7.4 beschrieben.

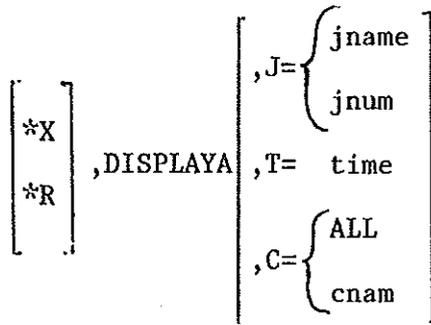
Sprache: VS/Assembler (siehe /3/ und /4/)

Linkage Editor Attribute: REUSABLE (siehe /11/)

Entrypoint: IATADIU

7.1 Aufruf von DISPLAYA

DISPLAYA wird über den JSS mittels folgender Kommandos (siehe /7/) von einer JES3-Konsole initialisiert, aktiviert oder beendet: *CALL, *RESTART oder *CANCEL.



*C, DISPLAYA

jname	Jobname
jnum	Jobnummer
time	Zeitintervall in Minuten
cnam	Jobklassenname

7.2 Erläuterungen zur Implementierung von DISPLAYA

Bei der Implementierung von DISPLAYA wurden folgende JES3-Kontrollblöcke und -Makros benutzt:

Kontrollblöcke: CNS, DSP, FCT, JCT, JDAB, JDS, JMR, JQE, JST, RQ, TVT
(siehe /9/ und Abb.6b)

Ein zusätzlicher MVS-Kontrollblock: CVT

Makros:

DSP-Programmverwaltung: ACALL,ARETURN
Spooldateiverwaltung: ABLOCK,ACLOSE,AGETBUF,ALOCATE,AOPEN,APURGE,
APUTBUF,ARELEASE,AWAIT,IATXRELC,JESREAD
Kommunikation: IATXSCN1,INTERCOM,LOGIN,LOGOUT,MESSAGE
Jobverwaltung: IATXJCT,IATXJQE
Speicherverwaltung: AGETMAIN,APUTMAIN
Ausgabeverwaltung: SPINOFF
Ressourcenkontrolle: AWAIT
weitere Makros: ATIME,IATXTOD

Die Register werden folgendermaßen benutzt,wobei die Konventionen für die DSP-Programmierung (siehe /3/) eingehalten werden.

R0,R1 Parameterregister
R2 Register für die Anfangsadresse der JCT
R10 Basisregister
R11 Register für die Anfangsadresse der FCT von DISPLAYA
R12 Register für die Anfangsadresse der TVT
R14,R15 Register für den Rücksprung aus einem Unterprogramm

7.3 Eingabeverarbeitung

Aus der Abb. 20 ist zu ersehen, wie DISPLAYA vom JES3 aufgerufen wird und wie die Eingabeparameter übergeben werden.

Erhält der Console Service die Kontrolle vom MFM, hängt er den Text der Nachricht an die Nachrichtenqueue der WTD-Driver FCT und 'posts' den WTD. Der WTD lädt die CALLDSP-Routine (IATGRCD). CALLDSP führt drei Gültigkeitsprüfungen durch:

- a) DSP-Name im DSP-Dictionary
- b) Ist das DSP callable, d.h. von einer Konsole aufrufbar.
- c) Anzahl der Zeichen des DSP-Names < 8

Falls alle Bedingungen erfüllt sind, werden vom CALLDSP für DISPLAYA folgende Kontrollblöcke in Puffern erstellt und in die Spooldatei geschrieben: JDAB, JMR, JCT, CNS. Der JCT-Entry für DISPLAYA wird an die JCT-Queue gehängt; der JSS wird 'geposted' (Zusammenhang der Kontrollblöcke in der Queue aus Abb. 6b ersichtlich).

Erhält der JSS vom MFM die Kontrolle, sucht er in der JCT-Queue nach einem JCT-Entry, den er bearbeiten kann. Für diesen JCT - im Beispiel DISPLAYA - bildet der JSS einen FCT- und RQ-Entry und hängt sie an die zugehörigen Queues, wobei in die FCT eine Einsprungsadresse für den JSS gespeichert wird.

Wenn der MFM die Kontrolle an die FCT von DISPLAYA gibt, findet über die obige Adresse ein Rücksprung in den JSS statt, der den DSP-Code mittels der JES3-Routine IATGRJR lädt und die Kontrolle an DISPLAYA übergibt.

*X, DISPLAYA, T=30

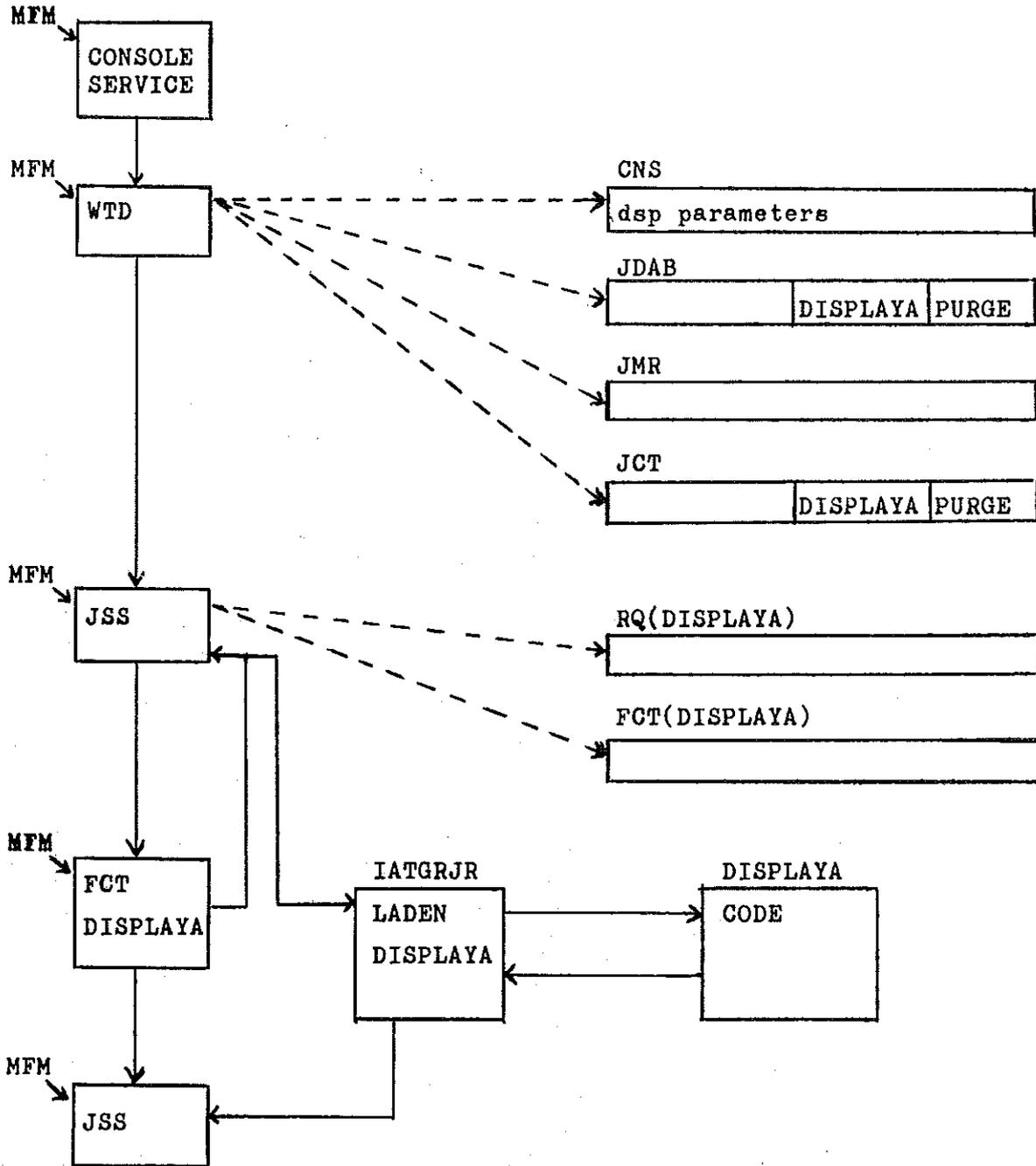


Abb. 20 DSP Scheduling

Im DISPLAYA wird der Eingabeparameterblock CNS mittels des Makros JESREAD von der Spooldatei in einen Puffer geholt, mit Hilfe des Makros IATSCN1 entschlüsselt und für die weitere Verarbeitung aufbereitet.

7.4 Funktionsbeschreibung

Die Hauptaufgabe von DISPLAYA besteht darin, Informationen über den Status der gesamten Jobqueue nach eines Ablauf vorgegebenen Zeitintervalls auf einem Drucker anzuliefern.

Es folgen Beispiele, wie diese Funktion von einer Konsole aus initialisiert wird.

a) *X,DISPLAYA oder *X,DISPLAYA,T=0

Dieses Konsolkommando erzeugt einen Ausdruck des Jobqueestatus zu dem Zeitpunkt, an dem das Kommando abgesetzt wird.

b) *X,DISPLAYA,T=30

Dieses Kommando liefert einen Ausdruck zu dem Zeitpunkt, an dem es abgesetzt wird, und dann jeweils in Intervallen von 30 Minuten. Diese Funktion wird auf folgende Weise realisiert. Nach dem Ausdruck des Jobqueestatus werden in DISPLAYA die Makros ATIME und AWAIT ausgeführt. Die Anwendung des AWAIT-Makros versetzt die FCT von DISPLAYA solange in den Wartezustand (siehe Abb. 20), bis dem DSP über das ATIME-Makro der Ablauf des vorgegebenen Zeitintervalls gemeldet wird. Die FCT wird 'geposted', und DISPLAYA erhält vom MFM wieder die Kontrolle.

Ist DISPLAYA erst einmal mittels eines *CALL-Kommandos wie in Beispiel 7.4b initialisiert worden, kann eine Änderung des Zeitintervalls nur mit dem folgenden Kommando durchgeführt werden (Begründung siehe 7.8), wobei wieder der Jobqueestatus ausgedruckt wird:

*R,DISPLAYA,T=90

Ein nach Beispiel 7.4b initialisiertes DISPLAYA kann durch folgende Konsolkommandos beendet werden:

c) *R,DISPLAYA,T=0 oder *C,DISPLAYA

Zwei weitere Aufgaben von DISPLAYA bestehen darin, einem Operator auf einer Konsole Informationen über den Status eines bestimmten Jobs oder über den Zustand der Job- und Ausgabe-Klassen zu vermitteln. Es folgen Beispiele der dafür notwendigen Konsolkommandos:

d) *X,DISPLAYA,C=A oder *R,DISPLAYA,C=A

Informationen über die Jobklasse A

*X,DISPLAYA,C=ALL oder *R,DISPLAYA,C=ALL

Informationen über die Verteilung der Jobqueue auf die Jobklassen

e) *X,DISPLAYA,J=ADIO22XX oder *R,DISPLAYA,J=ADIO22XX
*X,DISPLAYA,J=nnn oder *R,DISPLAYA,j=nnn

Informationen über einen bestimmten Job

Die *RESTART-Kommandos müssen immer dann gegeben werden, wenn DISPLAYA schon nach Beispiel 7.4b initialisiert worden ist.

7.5 Ermittlung der Job-Status Informationen

Alle Informationen, die DISPLAYA erfaßt, sind in JES3- und MVS-Kontrollblöcken enthalten, die entweder auf der Spooldatei stehen oder im Speicher resident sind.

Die im Speicher residenten Blöcke enthalten folgende Angaben:

TVT:

die Anfangsadresse der CVT und die Entrypointadressen der JES3-Makros

CVT:

das Datum; die Uhrzeit wird mit Hilfe des IATXTOD-Makros ermittelt.

JQE-Queue:

Diese Queue enthält für jeden Job im System einen Entry. Ein IATXJQE-Makroaufruf liefert die Anfangsadresse eines bestimmten JQE-Entry's. Dabei werden in DISPLAYA zwei Fälle unterschieden.

1. Fall (siehe 7.4e):

Mittels des eingegebenen Jobnamens bzw. der Jobnummer wird der zugehörige JQE-Entry ermittelt.

2. Fall (siehe 7.4a und 7.4d):

Die JQE-Queue, die nach Jobprioritäten geordnet ist, wird in diesem Fall auf Prioritätsleveln sukzessiv durchsucht, beginnend mit Priorität 14. Der Prioritätslevel 15 enthält JQE-Entries von Jobs wie Initiator, SYSLOG usw., die den Rechenzentrumsbenutzer nicht interessieren.

Der erste IATXJQE-Makroaufruf liefert die Anfangsadresse des ersten Jobs auf dem entsprechenden Prioritätslevel, wobei gleichzeitig für diesen Level ein Enqueue-Zustand erzeugt wird. Jeder weitere IATXJQE-Aufruf liefert, falls vorhanden, die Anfangsadresse des folgenden Jobs auf diesem Level.

Ist ein Prioritätslevel vollständig durchsucht, wird für diesen Level eine Dequeue-Operation ausgeführt. Dann wird die Priorität um Eins erniedrigt, und der nächste Level durchsucht usw., bis alle Prioritätslevel von 14 bis 1 abgearbeitet worden sind. Damit sind die JQE-Entries aller Jobs im System erfaßt. Wesentlicher Bestandteil eines JQE-Entry's ist die JCT-Adresse in der JCT-Datei.

JCT:

Mit Hilfe der obigen Adresse und des IATXJCT-Makros wird der zugehörige JCT-Entry in einen Puffer geholt. Sobald DISPLAYA Zugriff zu diesem Puffer erhält, wird der Inhalt dieses Puffers in einen Arbeitsspeicher, der von DISPLAYA mittels eines AGETMAIN-Makroaufrufs erzeugt wird, transferiert, um die Datensicherheit zu gewährleisten, falls eine andere JES3-Funktion gleichzeitig auf diesen Entry zugreifen sollte. Nach der Umspeicherung wird der JCT-Entry durch einen IATXJCT-Aufruf wieder freigegeben.

Der JCT-Entry im Arbeitsspeicher enthält folgende Informationen:

Jobname, Jobnummer, Jobpriorität, Jobklasse, Jobholdstatus
Jobnetzname, falls der Job ein Netzjob ist
geschätzte Anzahl der Druckzeilen (LINES-Parameter in der Job-Karte)
geschätzte Anzahl der zu stanzenden Karten (CARDS-Parameter)
Größe der VS-Region (REGION-Parameter)

Die JCT-Schedulerelemente sind Anhang eines JCT-Entry's. Ihnen wird der Status der Schedulerelemente CONVERTER/INTERPRETER, MAIN SERVICE und OUTPUT SERVICE entnommen.

Ebenfalls Bestandteil eines JCT-Entry's sind die Trackadressen der Kontrollblöcke JMR, JST und JDS auf der Spooldatei.

Diese Blöcke sind normalerweise nicht im Speicher resident und müssen mit Hilfe der obigen Adressen und JESREAD-Makroaufrufen von der Spooldatei in einen Puffer geholt werden. In diesen Blöcken sind folgenden Angaben enthalten:

JMR: Jobstartdatum und Jobstartzeit

JST: Volume Setup Anzahl und die Volume Serial Namen

JDS: Ausgabeklassennamen

Nachdem alle Informationen erfaßt worden sind, werden die von DISPLAYA angelegten Puffer mittels APUTMAIN-Aufrufen und die JES3-Kontrollblöcke mittels ARELEASE-/IATXRELC-Aufrufen freigegeben.

7.6 Ausgabeverarbeitung

Die Informationen werden in einem Ausgabepuffer gesammelt. Falls sie in den Kontrollblöcken binär gespeichert sind, werden sie vorher in die alphanumerische Darstellung umgewandelt. Bei der Ausgabe werden zwei Fälle unterschieden.

1. Fall (siehe 7.4d und 7.4e):

Der Inhalt des Ausgabepuffers wird mittels MESSAGE-Makroaufrufen auf die Konsole transferiert, deren Adresse im Kontrollblock CNS steht.

2. Fall (siehe 7.4b):

Mit Hilfe der Makros AOPEN und ACLOSE wird in der Spooldatei ein MRF erstellt. In diesen MRF wird der Inhalt des Ausgabepuffers mittels der Makros ALOCATE und ABLOCK gespeichert. Für diesen MRF wird dann ein JDS erzeugt, auf den das SPINOFF-Makro angewendet wird. Das hat zur Folge, daß der MRF auch dann auf einem Drucker ausgegeben wird, falls DISPLAYA nicht beendet wird wie im Beispiel 7.4b, in dem DISPLAYA in den Wartezustand versetzt wird. Zusätzlich läßt DISPLAYA in diesem Fall über einen INTERCOM-Aufruf das DSP IATUTDD den Zustand aller Jobnetze (siehe 7.8) ausdrucken.

Beispiele für die Druckausgabe von DISPLAYA sind aus den Abbildungen 21 - 23 zu ersehen.

7.7 Fehlerbehandlung und Nachrichten

Die Fehlerbehandlung im DISPLAYA besteht wesentlich darin, die Konsolkommandos, die DISPLAYA erhält (siehe 7.3), formal zu überprüfen, wie zum Beispiel:

Vollständigkeit des Kommandos

Gültigkeit der Schlüsselwörter (J= , C= , T=)

Anzahl der Zeichen des Jobnamens < 8 alphanumerische Zeichen

Anzahl der Zeichen der Jobnummer < 4 numerische Zeichen

Anzahl der Zeichen des Zeitintervalls < 4 numerische Zeichen

Gültigkeit des Jobklassennamens

Ist eine der obigen Bedingungen nicht erfüllt, wird auf der Konsole, auf der das Kommando eingegeben wurde, folgende Nachricht ausgedruckt:

```
IAT7760  INVALID PARAMETER , DISPLAYA CANCELLED
```

Eine Fehlernachricht erfolgt auch dann, falls der Name bzw. die Nummer eines Jobs formal richtig ist, der Job aber unter diesem Namen bzw. dieser Nummer nicht im System ist. Sie lautet:

```
DISPLAYA:  JOB  xxxxx  NOT IN SYSTEM
```

7.8 Environment von DISPLAYA

Um DISPLAYA im JES3 aktivieren zu können, muß im DSP-Dictionary ein Entry für DISPLAYA hinzugefügt werden; damit wird DISPLAYA von einer Konsole aus aufrufbar.

Ist DISPLAYA nach Beispiel 7.4b initialisiert worden, und folgen weitere Kommandos dieser Art, wird jedes Mal ein neues Exemplar von DISPLAYA in den Speicher geladen. Um das zu vermeiden, muß im Standard-Exit 18 ein Eingriff vorgenommen werden. Der Exit 18 erhält vom Console Service immer dann die Kontrolle, wenn von einer JES3-Konsole ein Kommando eingegeben wird. Dieser Exit wurde um folgende Abfrage erweitert: Ist DISPLAYA schon nach Beispiel 7.4b initialisiert? Falls ja, wird jedes weitere Kommando dieser Art zurückgewiesen.

Weitere Eingriffe wurden im DSP IATUTDD erforderlich (siehe 7.6). Dieses DSP liefert einen Ausdruck über den Zustand aller Jobnetze im System, wobei beim Ausdrucken für jedes Jobnetz eine neue Seite begonnen wird. Um diesen Ausdruck zu komprimieren, wurde der Seitenvorschub beseitigt.

Standardmäßig werden im Ausdruck der Jobnetze die Jobnamen ausgedruckt. Da die Jobnamen im System nicht eindeutig sind, wurden den Jobnamen die entsprechenden Jobnummern hinzugefügt.

DISPLAYA wird seit der Einführung von MVS/JES3 in der ADI eingesetzt.

```
+R DISPLAYA,C=ALL
JOB-CLASS   TOTAL NUMBER   NUMBER OF JOBS   NUMBER OF JOBS
              OF JOBS      (MAIN COMPLETED) (WITH SETUP)
G              003           001              001
N              001           000              001
S              003           000              002
```

Abb. 21 Konsolenausgabe nach Beispiel 7.4d

```
+R DISPLAYA,J=119
=====
10 0119 ADI022R5 START=11.00.54 LINES=014000 CARDS=00700 CLASS=G
MAIN=A OUT=N JOB-TIME=(0000,20)          SETUPS=01
INDEX=00 MDS ALLOC PROCESSING JSTAT=M168 ,RFQ RESOURCE NAVAIL
=====
```

Abb. 22 Konsolenausgabe nach Beispiel 7.4e

ABBREVIATION-GLOSSARY

DATE: 79.017 TIME: 11.04.32

```

=====
OUTS   OUTPUT-SERVICE   *STATUS-PARAMETER*   *HOLD-PARAMETER*
      (PRINT/PUNCH)     *(A)* ACTIVE          J JCL-HOLD
PRTY   PRIORITY         (C) COMPLETE         T TSO-HOLD
EST.   ESTIMATED       (N) NO STATUS        O OPERATOR-HOLD
=====
    
```

***** JOB QUEUE *****

```

=====
JOB  JOBNAME  JOB EST.  EST. NO.OF  JOB-CARD  HOLD  MAIN  OUTS  TIME-IN  DATE-IN  EST. CPU  OUTPUT-  JOB  JOB
NO.  PRTY  LINES  CARDS  SETUPS  REGION  HOLD  MAIN  OUTS  SYSTEM  SYSTEM  TIME  CLASSES  CLASS  NET-NAME
=====
0115 ADI022A3 10 007000 00900      0800K  J  (N)  (N)  11.00-53 79.017 (0045,00) A  S  NETZ2
0118 ADI022A2 10 014000 01100 01 0300K  *(A)* (N)  11.00-53 79.017 (0000,30) A  S
      TAPE-SETUP: DV1728
0119 ADI022R5 10 014000 00700 01 0100K  *(A)* (N)  11.00-54 79.017 (0000,20) A  G
      TAPE-SETUP: DV1725
0120 ADI022A5 10 005000 00900 01 0300K  *(A)* (N)  11.00-54 79.017 (0000,30) A  S
      TAPE-SETUP: DV1734
0121 ADI022M6 10 014000 01200      0120K  (C)  (A)  11.00-55 79.017 (0000,20) A,H  G
0122 ADI022R3 10 020000 00900      0120K  J  (N)  (N)  11.00-55 79.017 (0000,10) A  G  NETZ2
=====
    
```

*** NET-ID=NETZ2 *** 79.017 11:04:32

** JNCB PARAMETERS ** FLAG1=00 FLAG2=00 TOTAL COUNT=0002 COMPLETED=0000 PENDING=0000

JOB NAME	JOB NR.	STATUS	SUCCESSORS	H/R COUNT	ACTION PARAMETERS	ATTRIBUTES FLAG1	ATTRIBUTES FLAG2	ATTRIBUTES FLAG3	ATTRIBUTES FLAG4	FLAGS
ADI022A3	0115	IN NET HOLD		00001	NRML=D ABNML=R	00	00	0C	00	00
ADI022R3	0122	IN NET HOLD		00001	NRML=D ABNML=R	00	00	0C	00	00

Registerbenutzung:

R1,R2	Parameterregister
R10	Basisregister
R12	Register für die Anfangsadresse der TVT
R14,R15	Register für den Rücksprung aus CLCH

8.3 Eingabeverarbeitung

Der Aufruf von CLCH vom JES3 aus und die Übergabe der Eingabeparameter finden ebenso statt, wie es für DISPLAYA beschrieben wurde (siehe 7.3).

Der Eingabeparameterblock CNS, dessen Adresse im JDAB steht, wird in CLCH mittels des JESREAD-Makros in einen Puffer gelesen. Mittels der Makros IATXSCN1 und IATXSCN2 wird dieser Block entschlüsselt und für die Weiterverarbeitung aufbereitet.

8.4 Realisierung

Für einen Standardjob, der unter JES3 läuft (siehe Abb.3), ist eine Änderung seiner Jobklasse nur dann sinnvoll, falls er vom CONVERTER/INTERPRETER bereits bearbeitet, vom GMS aber noch nicht an das MVS zur Ausführung weitergeleitet worden ist. Begründung: Die Jobklassen werden erst im CONVERTER/INTERPRETER vergeben (siehe 3.0). Wird ein Job schon unter MVS ausgeführt, ist eine Änderung seiner Jobklasse nicht mehr möglich, da er vom GMS anhand seiner Jobklasse - d.h. der zugehörigen "performance group" - für das MVS selektiert wurde. Mit Hilfe der JCT-Schedulerelemente wird festgestellt, welches Standard-JES3-DSP den zu bearbeitenden Job augenblicklich kontrolliert.

Bei der Änderung der Jobklasse erhält der Job die Jobpriorität der neuen Klasse, falls nicht durch das Konsolkommando nach 8.1a oder 8.1b mit dem NP-Parameter eine abweichende Priorität spezifiziert wird. Um die Modifikation der Jobklasse wirksam werden zu lassen, müssen der Jobklassenname, die "performance"-Klassen/Gruppennummer und die Priorität in folgenden Kontrollblöcken geändert werden: JQE, JCT, JMR, RQ und NCB, falls der Job ein Netzjob ist.

Die JQE-Queue wird in derselben Weise, wie in 7.5 beschrieben, durchsucht, d.h. die Jobklassenmodifikation wird für einen oder mehrere Jobs (siehe 8.1a) oder für alle Jobs einer Klasse (siehe 8.1b) durchgeführt.

Die Adressen der Kontrollblöcke JCT und JMR werden, wie 7.5 dargestellt, ermittelt. Nach der Modifikation werden diese Kontrollblöcke mittels der Makros ARELEASE und AWRITE in die JCT- b.z.w. in die Spool-Datei zurückgeschrieben. Falls es sich um einen Netzjob handelt, muß die Priorität im zugehörigen NCB mittels der Makros JNCBHL, NCBTAFND, NCBTAPUT und JNCBREL, konsistent mit den anderen Kontrollblöcken, modifiziert werden, da sonst Systemfehler auftreten.

Ist kein RQ-Entry für zu ändernden Job vorhanden, d.h. die zu ändernde Jobklasse hat den Status OFF, sind nur die oben beschriebenen Modifikationen erforderlich. Gibt es einen RQ-Entry für den Job, müssen zwei Fälle unterschieden werden.

1. Fall:

Der Job wird von der MAIN DEVICE SETUP-Phase behandelt. Sein RQ-Entry hängt an einer der RQ-Queues, wie sie in Abb. 23 dargestellt sind. In diesem Fall können die Modifikationen im RQ-Entry sofort ausgeführt werden. Ein RQTAPUT-Makroaufruf ordnet den modifizierten RQ-Entry der neuen Priorität entsprechend ein.

2. Fall:

Der Job wird von der Jobselektionsphase des MAIN SERVICE (GMS) bearbeitet. Sein RQ-Entry hängt abhängig von der Jobklasse an einer der RQ-Queues, die Status "ready for execution" hat (siehe Abb. 24). In diesem Fall ist es erforderlich, daß der geänderte RQ-Entry vor Anwendung des RQTAPUT-Makros in CLCH von der Queue der alten Jobklasse zur Queue der neuen transferiert wird (siehe Abb. 25 und 26). Es ist notwendig, vor dem RQ-Entry-Transfer die Initiator der alten und neuen zugehörigen Jobgruppe mittels Intercom-Aufrufen zu stoppen, da bei einem Zugriff eines Initiators auf einen RQ-Entry während des Transfers Systemfehler auftreten können. Die Initiator der beiden Jobgruppen werden nach dem Umhängen des RQ-Entry's mit Hilfe eines weiteren Intercom-Aufrufs wieder gestartet.

8.5 Fehlerbehandlung und Nachrichten

Die Konsolkommandos, die CLCH erhält, werden auf formale Gültigkeit überprüft. Ist das Kommando fehlerhaft, wird auf der Konsole, auf der es eingegeben wurde, folgende Nachricht ausgegeben:

CONSOLE COMMAND INPUT ERROR

Ist ein Job unter dem eingegebenen Namen bzw. der Nummer nicht im System, oder ist eine der Bedingungen von Kapitel 8.4 nicht erfüllt, lautet die Fehlernachricht auf der Konsole (siehe Abb. 27 : Job ADI022M6 im OUTPUT SERVICE):

CHANGE TO CLASS A FOR JOB ADI022XX NOT POSSIBLE

Bei erfolgreicher Jobklassenänderung erscheint auf der Konsole die Nachricht (siehe Abb. 26):

JOB ADI022AA CLASS A CHANGED TO CLASS C

Jede Nachricht wird mittels des MESSAGE-Makros auf die entsprechende Konsole, deren Adresse dem JDAB entnommen wird, transferiert.

CLCH wurde am 15.2.78 dem Betrieb in der ADI zur Verfügung gestellt.

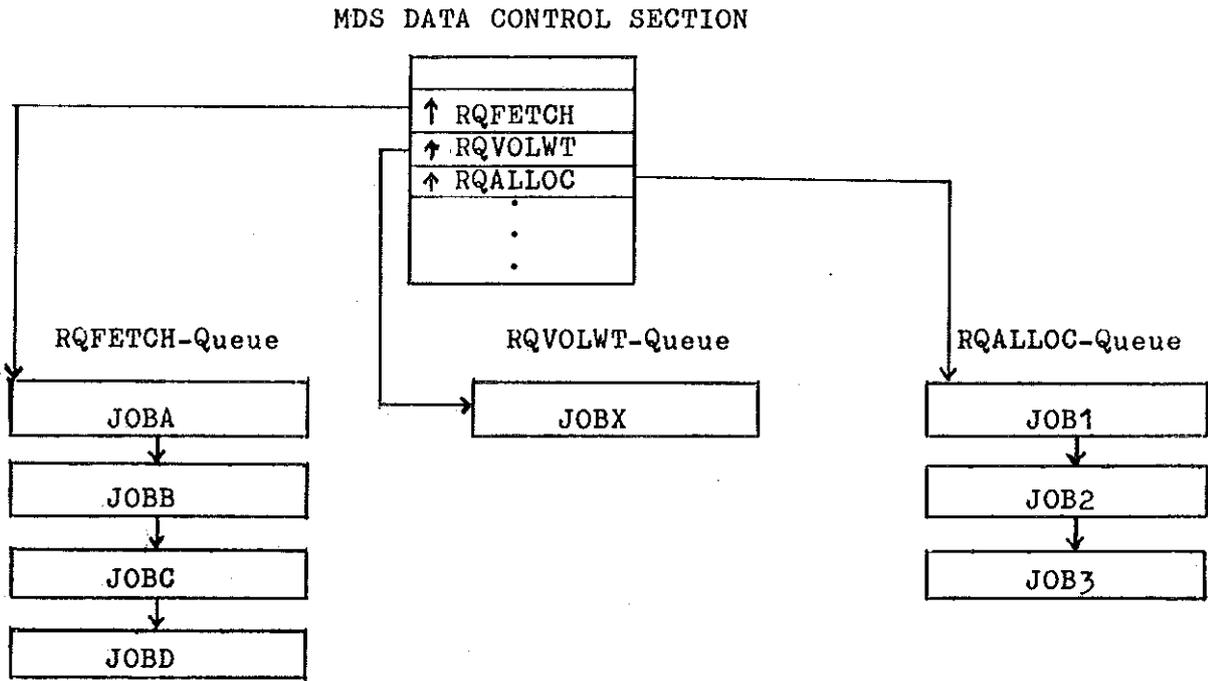


Abb. 23 RQ-Queues im MAIN DEVICE SETUP

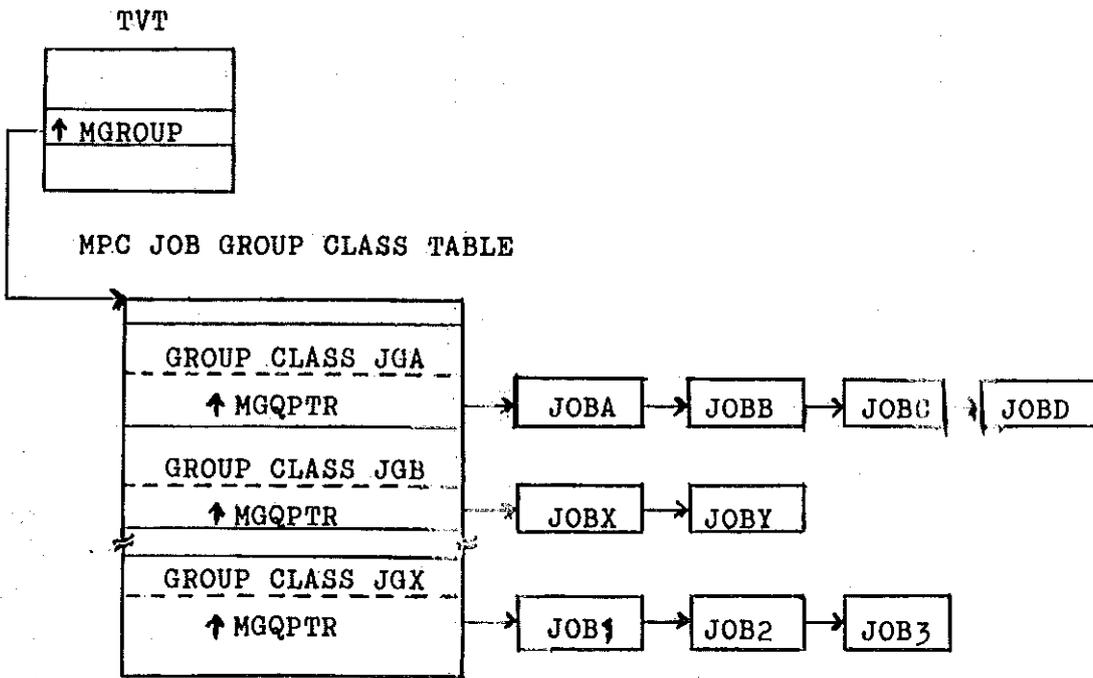
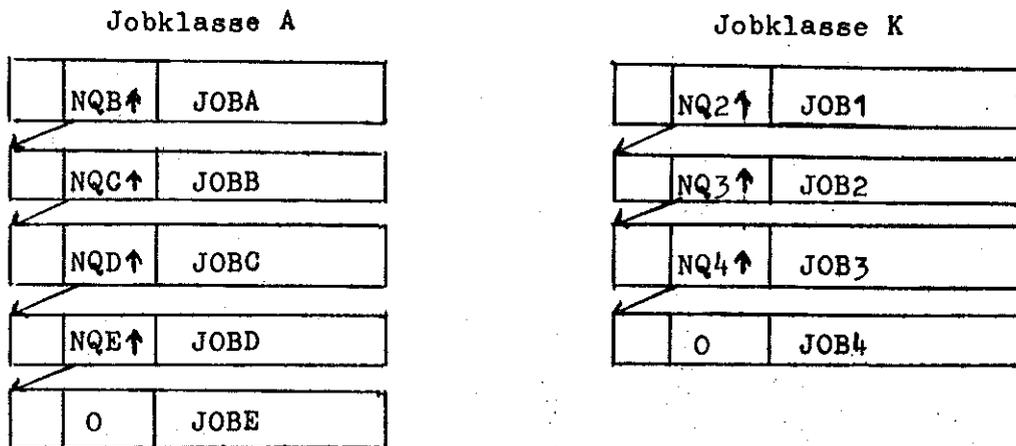


Abb. 24 RQ-Queues im GMS



NQ Anfangsadresse für den nächsten RQ-Entry

Abb. 25 Beispiel: RQ-Queue vor einer Jobklassenmodifikation (JOB2 von Klasse K nach Klasse A)

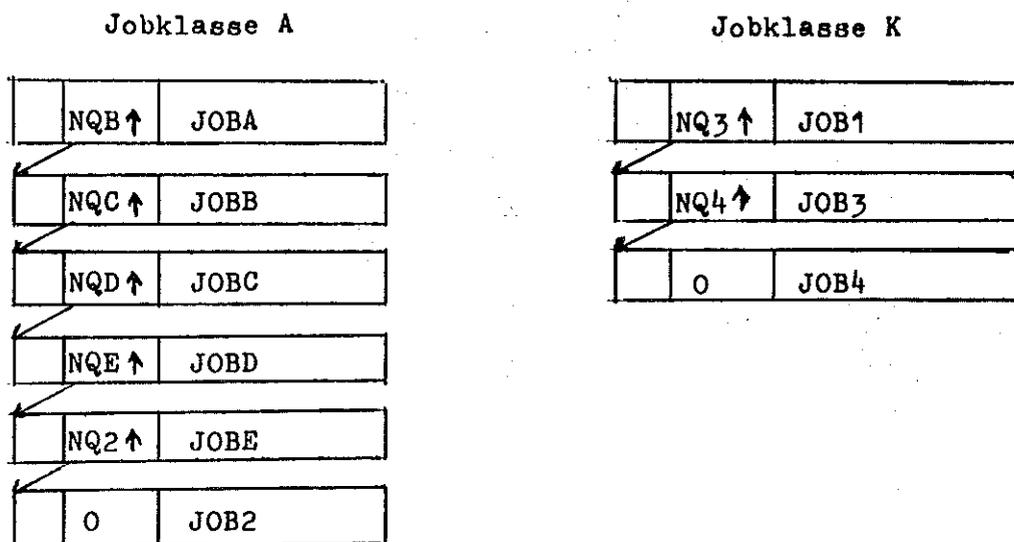


Abb. 26 Beispiel: RQ-Queue nach der obigen Jobklassenmodifikation

```
+X CLCH,C=G,NC=S
IAT6306 JOB 0129 IS CLCH      , CALLED BY CNB
JOB  ADI022A2  CLASS  G      CHANGED TO CLASS  S
JOB  ADI022R5  CLASS  G      CHANGED TO CLASS  S
JOB  ADI022A5  CLASS  G      CHANGED TO CLASS  S
CHANGE TO CLASS  S          FOR JOB  ADI022M6  NOT POSSIBLE
IAT7450 JOB 0129, CLCH      PURGED
```

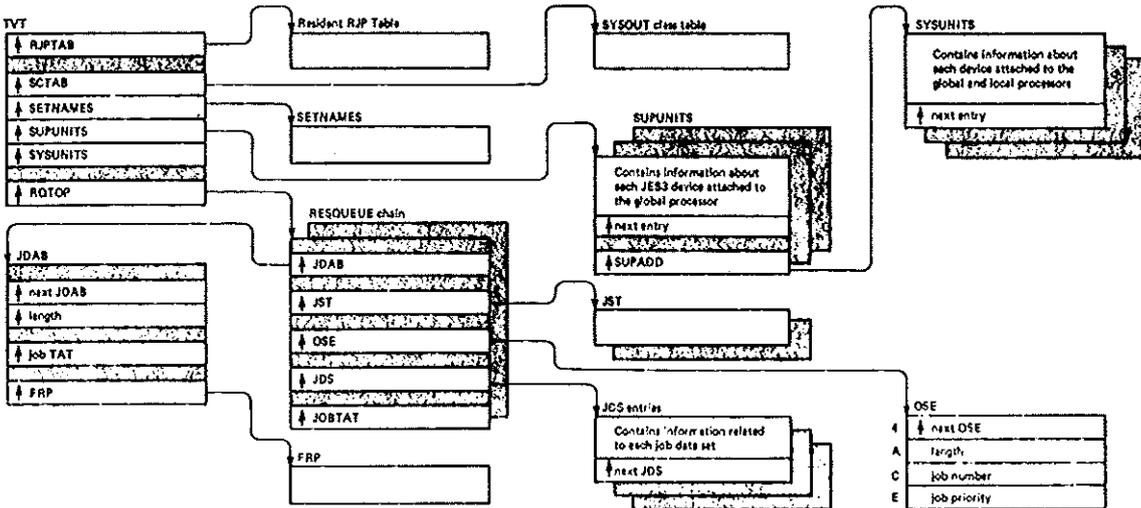
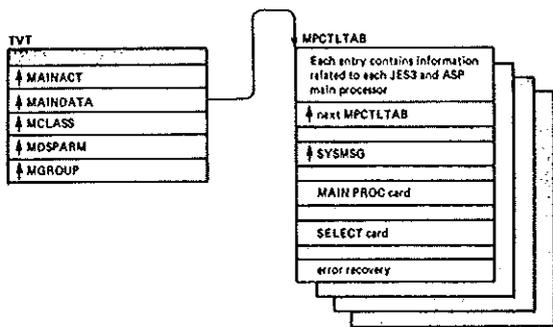
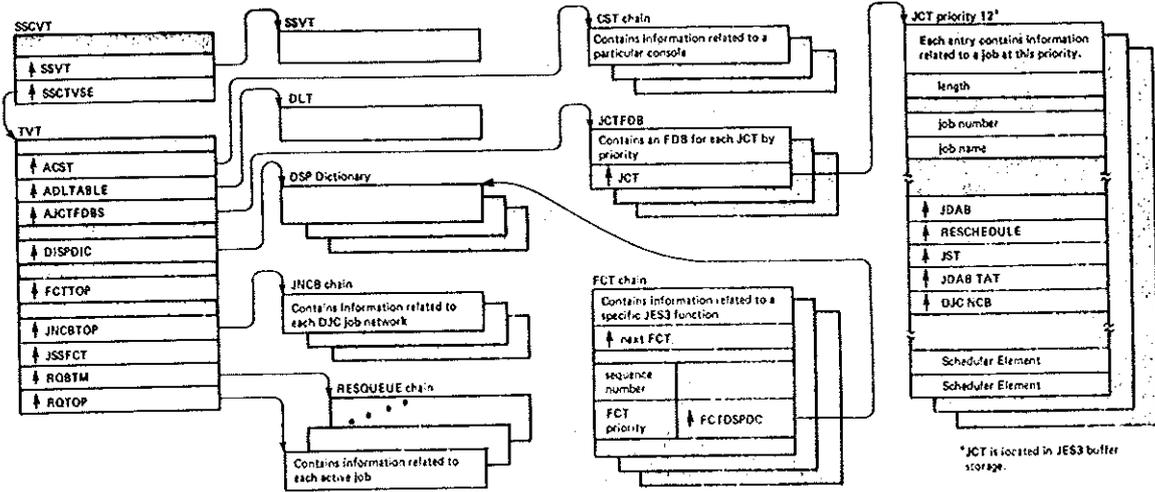
Abb. 27 Beispiel einer Konsolenausgabe von CLCH

9.0 Literatur-Hinweise.

- /1/ K.Jantzen, ASP - Ein Betriebssystem fuer Grossinstal-
lationen, IBM Nachrichten Nr.212,1972,Seite 343.
- /2/ H.W.Kohl, Verbundsysteme, IBM Nachrichten Nr.225,1975,
Seite 139.
- /3/ IBM Manual GC28-0607, Introduction to Jes3.
- /4/ IBM MANUAL GC28-0608, JES3.
- /5/ IBM Course BD43,MVS/JES3
- /6/ IBM Manual SY28-0612, Jes3 Logic.
- /7/ IBM Manual GC23-0008, MVS Jes3 Commands.
- /8/ IBM Manual GC28-0755, Initialization and Tuning Guide.
- /9/ IBM Manual GC28-0703, Jes3 Debugging Guide
- /10/ IBM Manual GC28-0750 OS/VS2 JCL
- /11/ IBM Manual GC26-3813 OS/VS Linkage Editor and Loader

10.0 Abkuerzungen und Begriffe.

CI	Converter/Interpreter. Jes3 Job-Segment, welches die JCL verarbeitet.
CPU	Central Processor Unit.
CTC	Channel-to-Channel Adapter. Geraet zur Verbindung von 2 Kanaelen von 2 verschiedenen Rechnersystemen.
CVT	Communication Vector Table.
DASD	Direct Access Spool Device.
DSP	Dynamic Support Program.
ECF	Event Control Flag.
FCT	Function Control Table.
GMS	Generalized Main Scheduling. Ein Algorithmus des Jes3, der entsprechend den Installationsparametern die Jobs zur Ausfuehrung auf den Main-Rechner selektiert.
I/O	Input/Output.
JCT	Job Control Table.
JDAB	Job Description and Accounting Block.
JDS	Job Dataset Control Block.
JMR	Job Management Record.
JQE	Job Queue Element.
JQX	JQE/JCT Access Control Table.
JSS	Job Segment Scheduler.
JST	Job Summary Table.
MEF	Multifunction Monitor. Der Jes3 Dispatcher
MCL	Main Processor Job Class Table.
MGP	Main Processor Job Class Group Table
MPC	Main Processor Control Table.
MVS	Multiple Virtual System.
NCB	Net Control Block.
RJP	Remote Job Entry.
RQ,RSQ	Resident Queue Element.
SPOOL	Simultaneous Peripheral Operation Offline.
SRM	System Resource Manager. Eine Anzahl von Program-Moduln des MVS, welche einen Algorithmus darstellen zwecks Ressourcen Verteilung und Kontrolle des MVS.
SSI	Subsystem Interface. Eine Anzahl von Programm-Moduln des Jes3, die eine 2-Weg Kommunikation des Jes3 oder anderen Subsystemen mit MVS Adressraeumen erlauben.
SVS	Single Virtual System.
SWA	Scheduler Work Area.
TCB	Task Control Table.
TVT	Transfer Vecctor Table.



Jes3 Control Blocks