Entire Operations GUI Client

Konzept und Leistungsumfang

Version 5.4.3

Dezember 2017

Dieses Dokument gilt für Entire Operations GUI Client ab Version 5.4.3.

Hierin enthaltene Beschreibungen unterliegen Änderungen und Ergänzungen, die in nachfolgenden Release Notes oder Neuausgaben bekanntgegeben werden.

Copyright © 2017 Software AG, Darmstadt, Deutschland und/oder Software AG USA, Inc., Reston, VA, USA, und/oder ihre Tochtergesellschaften und/oder ihre Lizenzgeber.

Der Name Software AG und die Namen der Software AG Produkte sind Marken der Software AG und/oder Software AG USA Inc., einer ihrer Tochtergesellschaften oder ihrer Lizenzgeber. Namen anderer Gesellschaften oder Produkte können Marken ihrer jeweiligen Schutzrechtsinhaber sein.

Nähere Informationen zu den Patenten und Marken der Software AG und ihrer Tochtergesellschaften befinden sich unter http://documentation.softwareag.com/legal/.

Diese Software kann Teile von Software-Produkten Dritter enthalten. Urheberrechtshinweise, Lizenzbestimmungen sowie zusätzliche Rechte und Einschränkungen dieser Drittprodukte können dem Abschnitt "License Texts, Copyright Notices and Disclaimers of Third Party Products" entnommen werden. Diese Dokumente enthalten den von den betreffenden Lizenzgebern oder den Lizenzen wörtlich vorgegebenen Wortlaut und werden daher in der jeweiligen Ursprungsprache wiedergegeben. Für einzelne, spezifische Lizenzbeschränkungen von Drittprodukten siehe PART E der Legal Notices, abrufbar unter dem Abschnitt "License Terms and Conditions for Use of Software AG Products / Copyrights and Trademark Notices of Software AG Products". Diese Dokumente sind Teil der Produktdokumentation, die unter http://softwareag.com/licenses oder im Verzeichnis der lizenzierten Produkte zu finden ist.

Die Nutzung dieser Software unterliegt den Lizenzbedingungen der Software AG. Diese Bedingungen sind Bestandteil der Produktdokumentation und befinden sich unter http://softwareag.com/licenses und/oder im Wurzelverzeichnis des lizenzierten Produkts.

Dokument-ID: OGC-ONOPCONCEPTS-543-20180305DE

Inhaltsverzeichnis

Konzept und Leistungsumfang v
I 1
1 Wozu dient Entire Operations? 3
Einleitung 4
Mehr als ein Scheduler 5
Zusammenfassung 8
2 Wozu dient der Entire Operations GUI Client? 9
II Entire Operations-Objekte 11
3 Beziehungen zwischen Entire Operations-Objekten 13
4 Eigentümer und Benutzerkennungen 15
Benutzer, Eigentümer und Job-Netzwerke 16
5 Job-Netzwerke 17
6 Jobs 19
Jobtypen 20
Job-Attribute 20
Jobs in einer Mehrrechnerkonfiguration 21
Jobkontroll-Besonderheiten 21
7 Logische Bedingungen 23
Übersicht 24
Eingabe- und Ausgabebedingungen 24
Jobs verknüpft durch Eingabe- und Ausgabebedingungen 25
Job-Netzwerk mit logischen Bedingungen 25
8 Mailboxen, Nachricht versenden 29
9 Ressourcen 33
10 Kalender 35
11 Zeitpläne 37
12 Symboltabellen und Symbole 39
13 Laufnummern 41
Laufnummern 42
Laufnummern-Bereich 42
Reservierte Laufnummern 42
14 Objekt-Versionierung 43
Nutzung des Versionierungskonzepts 44
Versionierung von Job-Netzwerken 44
Versionierung von Symboltabellen 49
III Entire Operations-Komponenten 55
15 Entire Operations-Komponenten 57
Master-Datenbank 58
Externe Jobkontrollanweisungen 59
Aktive Datenbank 59
Entire Operations-Monitor 61
Betriebssystem 61
Protokollierungsfunktionalität (Logging) 62

Berichtsfunktionalität (Reporting) 62
Editor 63
IV Entire Operations-Funktionen 65
16 Entire Operations-Funktionen 67
Entire Systems Management-Hauptbildschirm 68
Verwaltung der Job-Netzwerke 68
Zeitplanung für Job-Netzwerke 68
Job-Verwaltung 69
Job-Zeitplanung 69
Kalender-Definition 70
Verwaltung der Bedingungen 70
Job-Ende-Prüfung und -Aktionen 71
Verwendung von Ressourcen 73
Dynamische Generierung von Jobkontrollanweisungen (JCL) 73
Editor für System-Objekte 78
Berichtsfunktionen 79
Cross-Referenzen 80
Benachrichtigung 80

Konzept und Leistungsumfang

Wozu dient Entire Operations?	Vorstellung der besonderen Merkmale von Entire Operations und des Entire Operations GUI Client.
Wozu dient der Entire Operations GUI Client?	Verwendung des Entire Operations GUI Clients.
Entire Operations-Objekte	Die Benutzung von Entire Operations umfasst u.a. die Definition und Verwaltung bestimmter Objekte (Entitäten), welche in den folgenden Abschnitten näher beschrieben werden. Hierzu gehören: Eigentümer und Benutzerkennungen, Netzwerke, Jobs, logische Bedingungen, Mailboxen, Ressourcen, Kalender, Zeitpläne, Symboltabellen und Symbole.
Entire Operations-Komponenten	Entire Operations umfasst folgende Komponenten: die Master-Datenbank, die externen Jobkontrollanweisungen, die aktive Datenbank, den Monitor, das Betriebssystem (CPU), die Benutzerschnittstelle, die Protokollierungsfunktionalität („Logging“), die Berichtsfunktionalität („Reporting“), den Editor.
Entire Operations-Funktionen	Entire Operations bietet viele menü-unterstützte Funktionen, die Ihnen die Definition von Objekten innerhalb des Systems und die Kontrolle und Überwachung der Durchführung von Job-Netzwerken erlauben.

I

- 1 Wozu dient Entire Operations? 3
- 2 Wozu dient der Entire Operations GUI Client? 9

1 Wozu dient Entire Operations?

- Einleitung 4
- Mehr als ein Scheduler 5
- Zusammenfassung 8

Einleitung

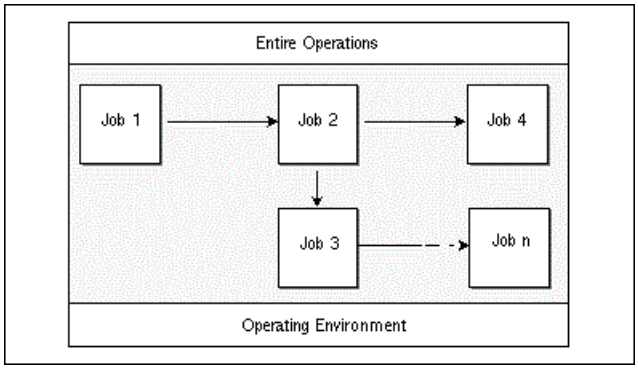
- Entire Operations
- Entire Operations GUI Client

Entire Operations

Entire Operations (Produktcode: NOP) ist das Softwaresystem der Software AG für die automatisierte Steuerung, Überwachung und Planung von **Job-Netzwerken**. Es bietet alle erforderlichen Funktionen zum Definieren beliebige Arten von Hintergrundverarbeitungen. Dabei wendet Entire Operations die Regeln an, die ihm vorher über eine komfortable Benutzungsoberfläche „beigebracht“ worden sind. Diese Benutzungsoberfläche besitzt alle Merkmale einer modernen, aufgabenorientierten Benutzerführung.

Zum Einsatz von Entire Operations sind keine Änderungen am Betriebssystem oder an einem der verwendeten Subsysteme erforderlich. Bestehende Jobkontrollanweisungen (JCL) können unverändert unter der Kontrolle von Entire Operations eingesetzt werden. Dadurch ist ein gleitender Übergang von bestehender Produktionssteuerung hin zu automatisierter Ablaufsteuerung möglich.

Entire Operations plant und überwacht Job-Netzwerke:



Bestehende Sicherheitssysteme wie RACF, ACF2, CA-TOP-SECRET oder SECOS werden unterstützt, das heißt, dort definierte Sicherheitsprofile finden auch bei der Arbeit mit Entire Operations Anwendung.

Zur Ausführung von Stapeljobs und Prozeduren benutzt Entire Operations definierte Schnittstellen zu den vorhandenen Spooling-Systemen bzw. funktional äquivalenten Teilen des Betriebssystems selbst.

Entire Operations GUI Client

Über den Entire Operations GUI Client (Produktcode: OGC) können PC-Benutzer aus einer Windows-Umgebung auf die Entire Operations-Objekte auf einem Großrechner zugreifen. Der Entire Operations GUI Client bietet eine grafische Benutzungsoberfläche: Job-Netzwerke können als Netzwerk-Grafik angezeigt und aus der grafischen Anzeige heraus verwaltet werden.

Mehr als ein Scheduler

Zusätzlich zu den Kernfunktionen eines Scheduling-Systems bietet Entire Operations folgende Besonderheiten:

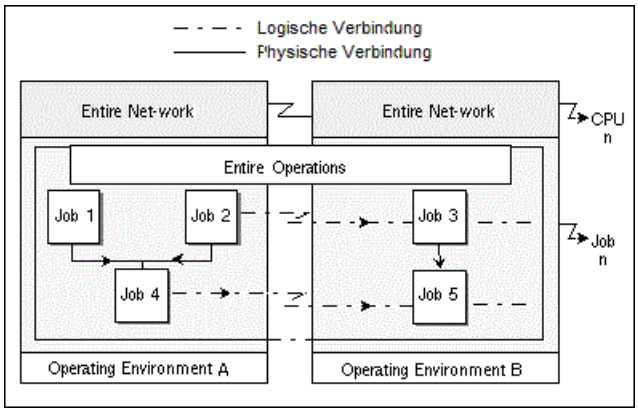
- Steuerung über Rechnergrenzen hinweg
- Entire Operations in einer Mehrrechnerkonfiguration
- Intelligente Prozesssteuerung
- Integrationsfähigkeit

Steuerung über Rechnergrenzen hinweg

- **Großrechnerumgebungen:**
  Entire Operations kann in allen TP-Umgebungen der Betriebssysteme z/OS, z/VSE und BS2000 eingesetzt werden, es unterstützt insbesondere Com-plete, CICS, TSO, IMS, *open*UTM und TIAM.
- **UNIX:**
  Entire Operations steuert Produktionsprozesse auf UNIX-Maschinen (HP-UX, SINIX RM, AIX), zum Beispiel, in Verbindung mit einer Großrechnerumgebung. Sie können aber auch Ihre Produktionsumgebung auf dem Großrechner von einem zentralen UNIX-Knoten kontrollieren oder aber Entire Operations benutzen, um einen Verbund von UNIX-Rechnern zu steuern. Wird Entire Operations in einer Mehrrechnerkonfiguration eingesetzt, bei der die einzelnen Rechnerknoten mit Entire Net-Work, der Kommunikationsdienstprodukt der Software AG verbunden sind, so kann Entire Operations auch hier Hintergrundprozesse überwachen und steuern. In einer solchen verteilten Umgebung können selbst einzelne Job-Netzwerke unter der Kontrolle von Entire Operations aus Verarbeitungsschritten bestehen, die auf verschiedenen Rechnerknoten ausgeführt werden.
- **Windows:**
  Entire Operations kontrolliert die Produktionsprozesse unter Windows.

Hintergrundprozesse, die in heterogenen Multi-CPU-Konfigurationen ablaufen, können ebenso von Entire Operations kontrolliert und beobachtet werden, wenn die Rechner mit dem Kommunikationsdienstprodukt Entire Net-Work der Software AG verbunden sind. In solchen verteilten Umgebungen können Job-Netzwerke aus Durchführungsschritten bestehen, die auf verschiedenen und selbst unterschiedlichen Rechnern ausgeführt werden.

Entire Operations in einer Mehrrechnerkonfiguration



Während der Entire Operations-Monitor die Verteilung und dezentralisierte Ausführung von Programmschritten überwacht, kann das verteilte Arbeiten trotzdem von einem einzigen Kontrollpunkt aus verfolgt und gesteuert werden.

Intelligente Prozesssteuerung

Jede Prozessautomatisierung hat die Notwendigkeit, die zukünftige Verarbeitung vorauszuplanen, um sie in der Sprache des Automationstools abzubilden. Leider unterscheidet sich manchmal die spätere Realität von der derzeitigen Planung. Nun kann man für solche Fälle natürlich Vorkehrungen treffen, die eine große Reihe von Ausnahmesituationen abdecken. Und natürlich bietet ihnen Entire Operations hierfür vielfältige Möglichkeiten, um auf solche Fehler gezielt reagieren zu können.

Aber dieser Prozess der gedanklichen Vorwegnahme möglicher Systemsituationen lässt sich nicht beliebig weit ausdehnen: Jeder neue Verarbeitungszweig besonders in umfangreichen Netzwerk-Topologien bedeutet eine Komplexitätssteigerung, die weder erwünscht noch notwendigerweise hilfreich sein dürfte.

Hierfür bietet Entire Operations die Möglichkeit an, die Verarbeitungsschritte selbst variabel zu gestalten. So können zum Beispiel die auszuführenden Jobkontrollanweisungen (JCL) dynamisch anhand aktueller Systemgegebenheiten (Plattenplatz, Inhalt von Warteschlangen im System, Vorhandensein von Dateien usw.) aufgebaut werden. Die Jobkontrollanweisungen „lernen“ also aus der augenblicklichen Situation und passen sich ihr an. Hierbei ist es sehr hilfreich, dass als Werkzeug die Softwareentwicklungsumgebung Natural zur Verfügung steht. Diese bietet einerseits eine ausdrucksstarke 4GL-Programmiersprache, um alle denkbaren Entscheidungskriterien abzubilden, andererseits stellt sie aber auch die benötigten Daten aufgrund vielfältiger Schnittstellen zu allen gängigen Datenhaltungs- und Betriebssystemen - auch in heterogenen Netzwerkumgebungen - zur Verfügung.

Somit ist es dann nicht mehr nötig, alle möglichen Problemsituationen vorwegzunehmen und bereits im Vorfeld feste Verarbeitungsschritte hierfür zu finden, sondern es müssen nur die Strategien (= Programme) zur Problemerkennung und -behebung vorgegeben werden.

Integrationsfähigkeit

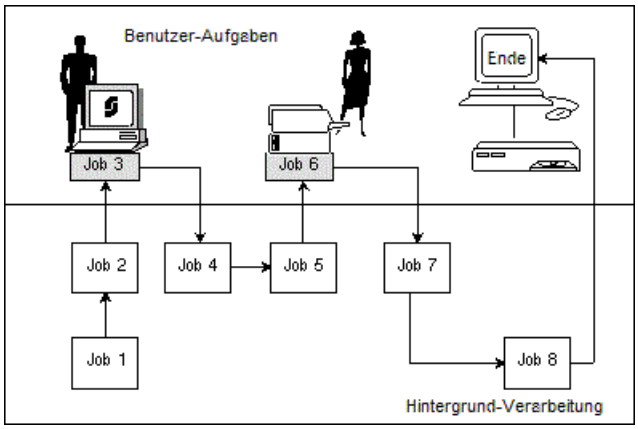
Integration anderer Anwendungen

Wer hat sich nicht schon darüber geärgert, dass man zur Erledigung seiner Arbeit mehrere Anwendungen oder Programmsysteme braucht. Hier waren dann oftmals langwierige (und langweilige) An- und Abmeldevorgänge vonnöten oder ein übergeordneter Session-Manager notwendig (was den Ressourcenverbrauch nicht unbedingt verminderte).

Entire Operations bietet die Möglichkeit, andere Anwendungen zu integrieren. Falls weitere Produkte der Entire System Management-Produktlinie installiert sind, werden diese erkannt und in das Hauptmenü von Entire Operations bzw. in die Baumstrukturansicht des GUI Clients aufgenommen. Das gilt zum Beispiel für:

- Entire Output Management
- Entire Event Management
- Entire System Server
- Natural ISPF

Einbeziehung von Fachpersonal in die automatisierte Ablaufsteuerung



Neben der zuvor beschriebenen eher programmtechnischen Integrationsfähigkeit bietet Entire Operations die Chance, alldiejenigen Personen am Produktionsprozess aktiv zu beteiligen, die das organisatorisch schon immer hätten tun sollen, aufgrund fehlender technischer Möglichkeiten aber bisher nicht konnten. Gemeint ist hier die Möglichkeit, über ein integriertes Mailbox-Konzept die „reine“ Hintergrundverarbeitung eines Schedulers mit Online-Benutzern zu verbinden.

Zu selbstdefinierten Zeitpunkten während der Verarbeitung von Hintergrundprozessen können Meldungen oder aber Eingabeaufforderungen an solche Mailboxen geschickt werden. Dies bewirkt einerseits, dass die Verarbeitung an diesem Punkt stoppt und dass andererseits alle mit dieser Mailbox verbundenen Benutzer davon in Kenntnis gesetzt werden. Diese können dann entsprechend reagieren, indem sie die geforderten manuellen Tätigkeiten ausführen (zum Beispiel, Papier

im Drucker nachfüllen) oder Variablenwerte eingeben, die für die weitere Verarbeitung benötigt werden. Nach Bestätigung der Eingabe wird die Verarbeitung fortgesetzt.

Mit dieser Technik kann z.B. erreicht werden, dass fachbereichsspezifische Informationen auch von Mitgliedern der Fachabteilungen eingegeben werden können, und dass trotzdem die Steuerung der gesamten Hintergrundverarbeitung unter zentraler Kontrolle bleibt.

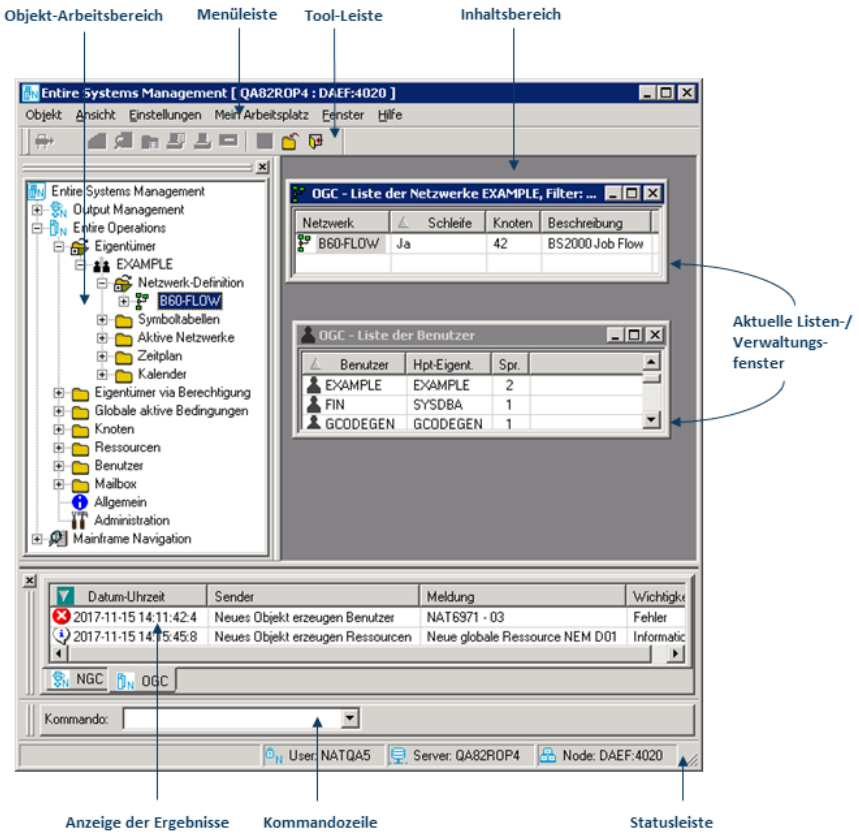
Zusammenfassung

Mit dem Einsatz von Entire Operations als automatisiertes Ablaufsteuerungssystem sichern Sie sich folgende wichtige Vorteile:

- transparente Unterstützung verschiedener Computer-Knoten, und dies sogar in heterogenen Umgebungen mit z/OS-, BS2000-, z/VSE- und UNIX-Plattformen,
- Verfügbarkeit in allen Host- und UNIX-Umgebungen, die von Natural unterstützt werden, zum Beispiel Com-plete, CICS, TSO, IMS, *open*UTM und TIAM,
- einfache Bedienung,
- wahlweise deutsche oder englische Benutzungsoberfläche,
- existierende Jobkontrollanweisungen laufen unverändert unter der Kontrolle von Entire Operations,
- keine Modifikationen am Betriebssystem,
- Einsatz dynamisch aufgebauter Jobkontrollanweisungen bzw. Prozeduren, dadurch Zugriff auf jegliche Informationen des Betriebssystems oder auf ein verfügbares Datenbanksystem zur Ausführungszeit der Jobnetze,
- Einbeziehung von Online-Benutzern in den Batch-Produktionsprozess durch das Konzept der **Mailboxen**,
- offene Schnittstelle für Benutzer-Anwendungen: Informationen aus Entire Operations können in jedes Produktionsprogramm integriert werden, Benutzer können andererseits ihren Input für die tägliche Produktion eingeben, auch für einige Tage im voraus.

2 Wozu dient der Entire Operations GUI Client?

Mit dem Entire Operations GUI Client (Produktcode: OGC) können Sie von einem PC aus die Entire-Operations-Funktionen auf einem Großrechner oder einer-UNIX-Plattform ausführen.

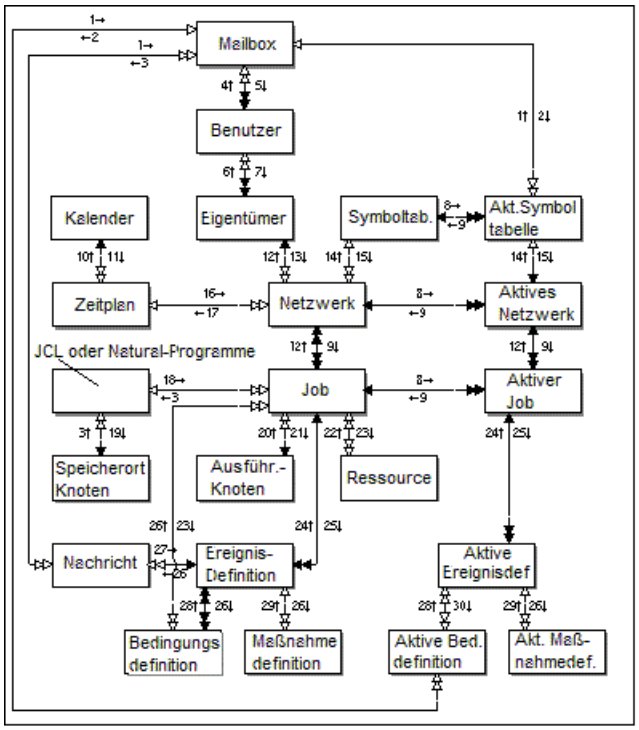


Weitere Informationen siehe *Elemente des Entire Systems Management-Hauptbildschirms* im *Benutzerhandbuch*.

II Entire Operations-Objekte

Beziehungen zwischen Entire Operations-Objekten
Eigentümer und Benutzerkennungen
Job-Netzwerke
Jobs
Logische Bedingungen
Mailboxen, Nachricht versenden
Ressourcen
Kalender
Zeitpläne
Symboltabellen und Symbole
Laufnummern
Objekt-Versionierung

3 Beziehungen zwischen Entire Operations-Objekten



1 = wird gesendet an
2 = enthält Aufforderung für
3 = enthält
4 = verwendet/kann verwenden
5 = enthält Nachricht für
6 = erteilt Berechtigung für
7 = kann auswählen
8 = verleiht Eigenschaften an
9 = besteht aus
10 = basiert auf
11 = ist Basis für
12 = gehört zu
13 = ist Eigentümer von
14 = erhält Parameter von
15 = enthält Werte für
16 = plant
17 = wird geplant auf Basis von
18 = wird ausgeführt als
19 = ist gespeichert auf
20 = ist Plattform für
21 = läuft auf
22 = ist Vorbedingung für
23 = ist abhängig von/ kann abhängig sein von
24 = bestimmt Ergebnis von
25 = wird geprüft auf
26 = triggert
27 = wird gesetzt gemäß
28 = wird gesetzt/zurückgesetzt gemäß
29 = wird ausgeführt gemäß
30 = setzt

Bedeutung der Pfeil-Symbole:

- darf nur eins sein
- keines oder höchstens eins
- keines oder eins oder viele
- muss mindestens eins sein, kann mehr als eins sein

4 Eigentümer und Benutzerkennungen

- Benutzer, Eigentümer und Job-Netzwerke 16

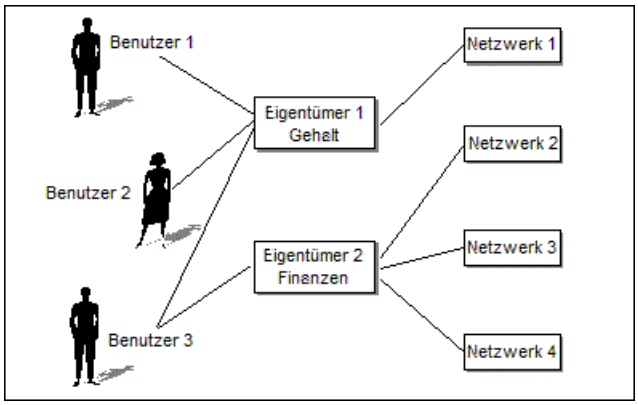
Jeder Benutzer von Entire Operations erhält eine eindeutige Benutzerkennung („UserId“), die für Sicherheitsprüfungen, Profil-Definitionen, Benachrichtigungen und Protokollierung herangezogen wird.

Jede Benutzerkennung ist mit einem Benutzerprofil verknüpft, das die Autorisierungen regelt. Benutzerprofile können von dazu berechtigten Benutzern (z.B. dem Systemverwalter) verändert werden.

Eigentümerkennungen werden verwendet, um einerseits Sicherheitsaspekte zu implementieren und andererseits Benutzer gruppieren zu können. Letzteres dient dazu, Verbindungen zu Job-Netzwerken zu vereinfachen. Ein Benutzer kann autorisiert werden, mehrere Eigentümerkennungen zu verwenden: ein Wechsel der Eigentümerkennung bedeutet dann, eine andere Menge von Job-Netzwerken zu selektieren, die in einem zweiten Schritt verwaltet werden können.

Benutzer, Eigentümer und Job-Netzwerke

Die folgende Abbildung zeigt ein Beispiel für die Verbindungen zwischen Benutzern, Eigentümern und Job-Netzwerken:



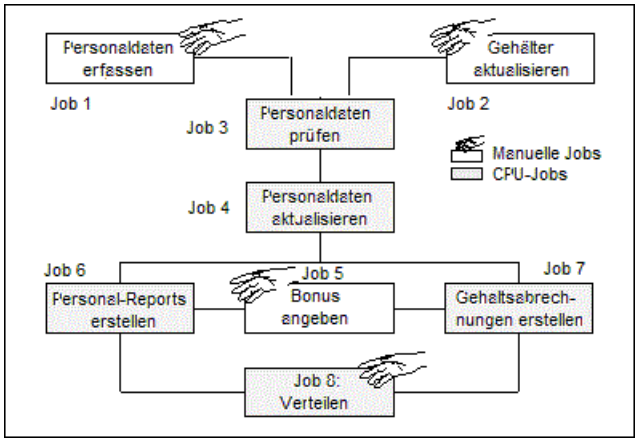
Jeder der drei dargestellten Benutzer kann alle Netzwerke verwalten, die dem Eigentümer „Gehalt“ (Payroll) gehören, darüber hinaus ist Benutzer 3 für alle Netzwerke autorisiert, die dem Eigentümer „Finanzen“ (Finance) zugeordnet sind.

Weitere Informationen siehe *Entire Operations-Benutzerkennung*, *Betriebssystem-Benutzerkennungen* und *Eigentümer* im *Benutzerhandbuch*.

5 Job-Netzwerke

Innerhalb Entire Operations umfaßt ein Job-Netzwerk eine Gruppe von Jobs, Tasks, Skripts oder Prozessen, die in Verbindung stehen können (aber nicht müssen) und die entsprechend eines Netzwerk-Zeitplans für die Abarbeitung aktiviert werden. Demnach kann ein Job-Netzwerk eine beliebige Einheit innerhalb des Produktionsablaufs darstellen.

So können in einen Job-Netzwerk auch manuelle Tätigkeiten eingebaut werden, die zu bestimmten Zeitpunkten vom Personal im Rechenzentrum oder aber der Fachabteilung ausgeführt werden sollen. Die folgende Abbildung veranschaulicht dies am Beispiel eines Job-Netzwerkes, so wie es in einer Lohnbuchhaltung eingesetzt sein könnte. Es umfaßt die automatische Erstellung von Rechnungsbelegen und Abstimmlisten:



Im Normalfall besteht ein Job-Netzwerk aus einer Folge von Jobs, die durch bestimmte Abhängigkeiten untereinander verbunden sind (z.B.: Wenn Job 1 OK geendet hat, starte Job 2). Diese Abhängigkeiten werden durch so genannte „logische Bedingungen“ dargestellt.

Ein Job-Netzwerk ist die kleinste Einheit, die von Entire Operations automatisch aktiviert werden kann. Durch die Verwendung von Kalendern können Job-Netzwerke automatisch vom Entire

Operations-Monitor-Programm aktiviert werden. Manuelle Aktivierungen können von jedem dafür autorisierten Benutzer vorgenommen werden.

Mehrere aktive Kopien (=Aktivierungen) des gleichen Netzwerkes können zu einem Zeitpunkt parallel verarbeitet werden, weil Entire Operations jede dieser Kopien durch eine eindeutige Laufnummer identifiziert. Diese Laufnummer wird jedem Netzwerk bei dessen Aktivierung automatisch zugeordnet.

Weitere Informationen siehe:

- ***Versionierung von Job-Netzwerken***
- *Job-Netzwerk* und *Netzwerk-Verwaltung* im *Benutzerhandbuch*.

6 Jobs

- Jobtypen 20
- Job-Attribute 20
- Jobs in einer Mehrrechnerkonfiguration 21
- Jobkontroll-Besonderheiten 21

Der Job stellt einen der zentralen Objekttypen innerhalb des Entire Operations-Systems dar. Ein Job im Sinne von Entire Operations ist eine benutzerdefinierte Aufgabe, die durch JCL-Anweisungen und Job-IDs, je nach Betriebssystem unterstützte Scripts oder Dateien, Entire Operations-Unternetzwerke oder Dummy-Jobs oder Natural-Programme ausgeführt wird.

Weitere Informationen siehe *Job* und *Job-Verwaltung* im *Benutzerhandbuch*.

Jobtypen

Entire Operations unterstützt folgende Jobtypen:

- Standard-Jobs des Betriebssystems (z/OS, z/VSE, BS2000)
- Gestartete Tasks (z/OS)
- Standard-Shell-Prozeduren des UNIX-Betriebssystems
- BAT-Dateien auf Windows-Systemen
- Andere Scripting Umgebungen unter UNIX und Windows (z.B.: Perl, Windows Scripting Host)
- Kommandozeilenorientierte ausführbare Programme unter UNIX und Windows
- FTP-Jobs
- SAP-Jobs
- Natural-Programme
- Datei-Generierung
- Windows Services

Weitere Informationen über Jobtypen siehe Abschnitt *Jobtypen und Job-Ausführungsmerkmale* im *Benutzerhandbuch* .

Daneben gibt es für nicht-CPU basierte Jobs noch den Typ des Dummy Jobs, der es erlaubt, Zeitfenster für nicht-CPU-basierte Jobs darzustellen oder beliebige Boolesche Verknüpfungen von Einzelbedingungen zu realisieren.

Job-Attribute

Jeder Job eines Netzwerks wird durch eine Reihe von Attributen identifiziert:

- (Logischer) Name
- Jobtyp
- Knotenkennung für den Job-Speicherort

- Knotenkennung für die Job-Ausführung
- Startzeit
- Job-Ende-Informationen
- Benutzerkennung, unter der der Job ausgeführt wird.

Ein solcher Job kann auch in mehreren Job-Netzwerken enthalten sein.

Jobs in einer Mehrrechnerkonfiguration

Wird Entire Operations in einer Mehrrechnerkonfiguration eingesetzt, kann der Speicherort eines Jobs (d.h. seines Inhalts) vom Ort seiner Ausführung abweichen: zur Ausführungszeit liest Entire Operations die Job-Informationen vom Speicherort und führt den Job auf dem Zielknoten aus.

Jobs in Netzwerken können durch so genannte *logische Bedingungen* untereinander verbunden sein.

Siehe auch *z/OS: JES2 /*ROUTE Statement*.

Jobkontroll-Besonderheiten

Dieses Abschnitt behandelt folgende Themen:

- z/OS: JES2 /*ROUTE Statement
- UNIX

z/OS: JES2 /*ROUTE Statement

Falls eine z/OS JES2 JCL folgendes Statement enthält, wird der Job auf der Ziel-Maschine ausgeführt:

```
/* ROUTE XEQ <target>
```

Solange wie die SYSOUT-Datei an die auftraggebende Maschine zurückgegeben wird, ist der laufende Job nicht zugänglich. Entire Operations stellt fest, dass eine Rückgabe vorliegt, und verhält sich bei solchen Jobs anders.

Bei zurückgegebenen z/OS-Jobs sind einige Merkmale außer Funktion, z.B.:

- Direkte Nachverfolgung der Ausführung.
- Auflisten von SYSOUT während der Job-Ausführung.
- Abbrechen.

Jedoch wenn die SYSOUT-Datei wieder verfügbar ist, können alle Job-Ende-Prüfungen und alle Job-Ende-Aktionen durchgeführt werden.

UNIX

Um die Datei .profile zu benutzen

Ausführung von UNIX Shell Scripts: Um die Verwendung von Profilen in Non-login Scripts (weil sie von Entire Operations gestartet werden) zu ermöglichen, werden Profile wie folgt behandelt:

1 Beim Starten eines UNIX Shell Scripts prüft der Entire Operations-Monitor, ob das Symbol `ETC-PROFILE` in der Symboltabelle des aktiven Jobs oder in einer anderen Symboltabelle in der Standard-Symbol-Suchhierarchie, bis hin zu `SYSDBA/A`, vorhanden ist.

   - Wird das Symbol `ETC-PROFILE` gefunden und enthält es ein "Y", dann benutzt das Batch Frame (`*.BF`) Script die */etc/profile and /etc/profile.local* als Source (nur falls sie vorhanden sind).
   - Das Benutzer-Script (`*.B`) findet die zurzeit gesetzten Umgebungsvariablen des Profil-Scripts.

2 Beim Starten eines UNIX Shell Scripts prüft der Entire Operations-Monitor, ob das Symbol `ENV` in der Symboltabelle des aktiven Jobs oder in einer anderen Symboltabelle in der Standard-Symbol-Suchhierarchie, bis hin zu `SYSDBA/A`, vorhanden ist.

   - Wird das Symbol `ENV` gefunden und ist es nicht leer, dann wird angenommen, dass es sich bei seinem Inhalt um ein Startup Script wie `$HOME/.profile` handelt.
   - Ist das Startup Script vorhanden, wird sein Name durch das Batch Frame (`*.BF`) Script den Umgebungsvariablen `ENV` und `BASH_ENV` zugewiesen.
   - Ist das Benutzer-Script (`*.B`) ein Bourne Shell, Bourne Again Shell, oder Korn Shell Script, dann führt die Shell den Inhalt der zuvor gesetzten Umgebungsvariable `ENV` bzw. `BASH_ENV` aus.

Anmerkung: Der Benutzer muss dafür sorgen, dass die verwendeten Profil-Scripts gegen Mehrfachausführung sind, z.B. indem er die Variable `PROFILEREAD` wie in Linux verwendet.

7 Logische Bedingungen

- Übersicht 24
- Eingabe- und Ausgabebedingungen 24
- Jobs verknüpft durch Eingabe- und Ausgabebedingungen 25
- Job-Netzwerk mit logischen Bedingungen 25

Weitere Informationen siehe *Logische Bedingungen*, *Eingabebedingungen* und *Job-Ende-Prüfungen und -Aktionen* im *Benutzerhandbuch*.

Informationen zur Prüfung von Bedingungen, siehe den Abschnitt *Prüfung von Bedingungen* im *Benutzerhandbuch*.

Übersicht

Die Verwendung logischer Bedingungen ist das zentrale Konzept von Entire Operations. Logische Bedingungen werden benutzt, um Abhängigkeiten zwischen Jobs oder Job-Netzwerken zu beschreiben. Solche logischen Bedingungen können entweder durch CPU-basierte Ereignisse oder aber durch manuelle Eingaben gesetzt werden. Diese Ereignisse müssen also eingetreten sein, bevor Entire Operations mit weiteren Schritten fortfahren kann.

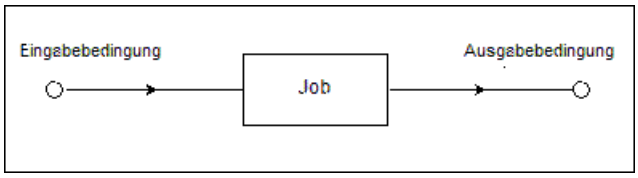
Sobald ein Job-Netzwerk aktiviert wird, erhält jede logische Bedingung eine Laufnummer. Mit ihrer Hilfe kann Entire Operations zwischen gleichen Ereignissen in unterschiedlichen Netzwerk-Aktivierungen unterscheiden.

Logische Bedingungen können verwendet werden als:

- Eingabebedingungen
- Ausgabebedingungen

Die folgende Abbildung veranschaulicht das Prinzip der Eingabe- und Ausgabebedingungen im Verhältnis zu einem Job:

Eingabe- und Ausgabebedingungen



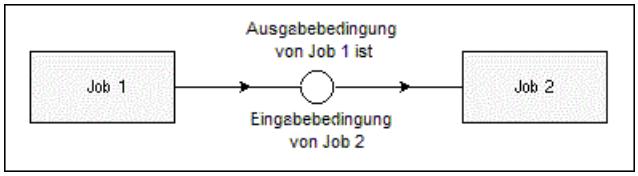
Alle Eingabebedingungen müssen erfüllt sein, bevor ein Job tatsächlich gestartet werden kann (Voraussetzung). Sie können eine beliebige Anzahl von Eingabebedingungen für einen Job definieren.

Eine Ausgabebedingung dagegen kann gesetzt oder aber zurückgesetzt werden, entsprechend dem Ergebnis vordefinierter Ereignisse. Diese Ereignisse sind entweder automatisch durch Entire Operations gegeben, sie können aber auch vom Benutzer definiert werden. Als Bestandteil der Job-Ende-Untersuchung überprüft Entire Operations das Vorhandensein solcher Ereignisse. Für

jedes dieser Ereignisse können dann auf Job- oder aber Job-Step-Ebene mehrere Ausgabebedingungen gesetzt oder zurückgesetzt werden.

Jobs verknüpft durch Eingabe- und Ausgabebedingungen

Jobs in Job-Netzwerken werden verknüpft, indem die Ausgabebedingung des einen Jobs als Eingabebedingung des nächsten Jobs definiert wird. Dies wird in folgender Abbildung veranschaulicht:

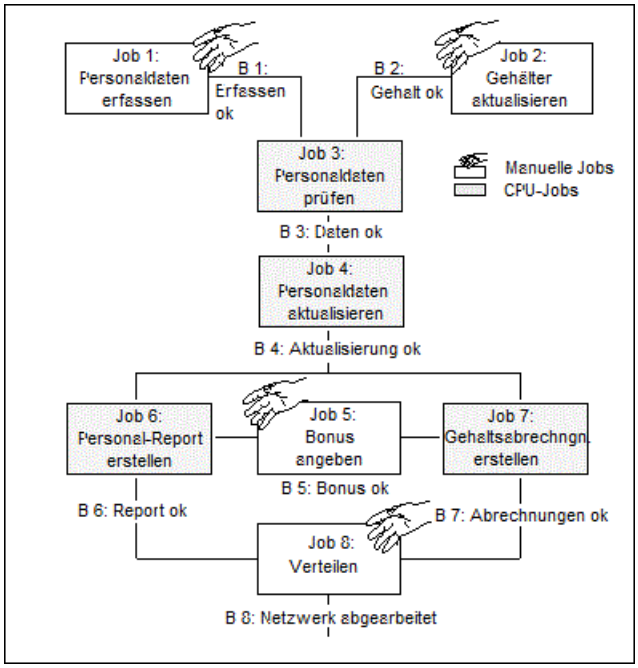


Tritt nun ein bestimmtes Ereignis als Ergebnis von Job 1 ein, so setzt dies die entsprechende Bedingung und signalisiert Entire Operations, dass Job 2 gestartet werden kann.

Sie können auch Jobs miteinander verknüpfen, die unterschiedlichen Job-Netzwerken angehören oder die auf verschiedenen Rechnerknoten ausgeführt werden.

Job-Netzwerk mit logischen Bedingungen

Die folgende Abbildung zeigt Job-Abhängigkeiten am Beispiel eines Netzwerks für die Lohnbuchhaltung:



Die folgende Tabelle gibt einen Überblick über die Job-Abhängigkeiten (logische Bedingungen), durch die Jobs in obiger Abbildung miteinander verknüpft sind.

Jobverknüpfung durch Eingabe- und Ausgabebedingungen

Job	Eingabebedingung (B)	Ausgabebedingung (B)
Job 1: Personaldaten erfassen		B 1: Erfassen - ok
Job 2: Gehälter aktualisieren		B 2: Gehalt - ok
Job 3: Personaldaten prüfen	B 1: Erfassen - ok	
	B 2: Gehalt - ok	B 3: Daten - ok
Job 4: Personaldaten aktualisieren	B 3: Daten - ok	B 4: Aktualisierung - ok
Job 5: Bonus angeben		B 5: Bonus - ok
Job 6: Personal-Reports erstellen	B 4: Aktualisierung - ok	
	B 5: Bonus - ok	B 6: Report - ok
Job 7: Gehaltsabrechnungsbelege erstellen	B 5: Bonus - ok	
	B 4: Aktualisierung - ok	B 7: Belege - ok
Job 8: Verteilen	B 6: Report - ok	
	B 7: Belege - ok	B 8: Netzwerk abgearbeitet

Zum Beispiel wird Entire Operations den Job 6 (Personal-Reports erstellen) solange nicht starten, bis die Eingabebedingungen B 4 und B 5 erfüllt sind. Diese sind auch als Ausgabebedingungen für die Jobs 4 bzw. 5 definiert.

Ein solcher Jobfluss ist vollkommen unabhängig von den Betriebssystem-Plattformen, auf denen die einzelnen Verarbeitungsschritte ablaufen.

8 Mailboxen, Nachricht versenden

Innerhalb Entire Operations dienen als Mailbox bezeichnete elektronische Briefkästen dazu, mit Netzwerken verknüpfte Meldungen und Aufforderungen an Benutzer und/oder Gruppen von Benutzern zu empfangen. Diese Meldungen können benutzt werden, um Benutzer über den aktuellen Status eines Netzwerks zu informieren oder aber die Eingabe von Daten anzufordern, die für die weitere Ausführung benötigt werden.

Unter solchen Meldungen bzw. Aufforderungen werden zusammengefaßt:

- Benutzerdefinierte Meldungen;
- Systemmeldungen des Entire Operations-Monitor-Programms;
- logische Bedingungen betreffende Eingabe-Aufforderungen, die von bestimmten Benutzern bestätigt werden müssen. Diese Bedingungen sind als von einer manuellen Tätigkeit abhängig definiert worden;
- Aufforderungen für noch offene Symbol-Eingaben, die zu zukünftigen Netzwerk-Aktivierungen gehören.
- Mit Hilfe des Konzepts der Mailboxen kann Entire Operations nicht-CPU-basierte Prozesse auf die gleiche Weise behandeln wie CPU-basierte:
  - sie können von logischen Bedingungen abhängig gemacht werden und selbst solche setzen;
  - durch die Zuordnung einer Mailbox zu diesen logischen Bedingungen kann festgelegt werden, wer über diese Bedingung informiert werden soll.

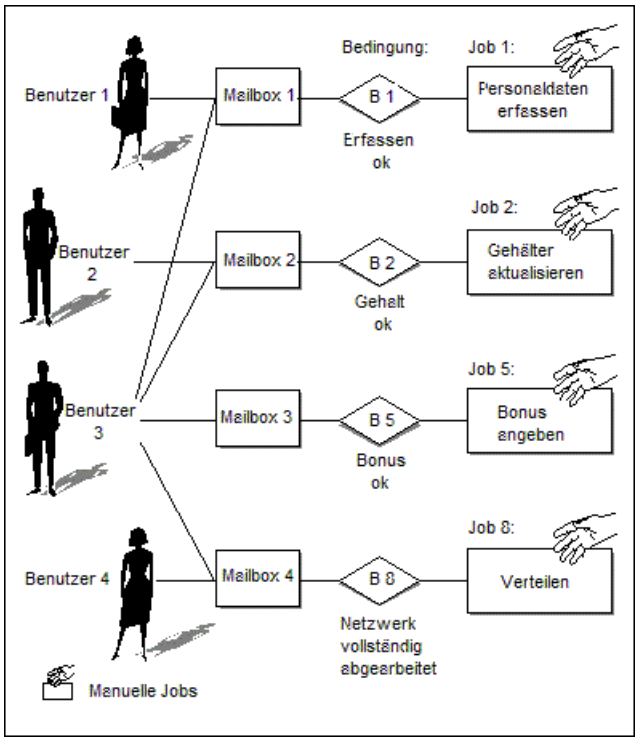
Jeder Benutzer, der mit einer Mailbox verknüpft ist, wird deren Liste der ausstehenden Meldungen und Aufforderungen erhalten. Hierzu muss er lediglich das Direktkommando `MAIL` eingeben. Durch einen einzigen Tastendruck kann der Benutzer dann die Bedingung als erfüllt markieren oder andere der geforderten Tätigkeiten durchführen, wie z.B. die Werteingabe für Symbole. Danach verschwindet diese Meldung von der Liste, das Entire Operations-Monitor-Programm wird über die erfolgte Bestätigung automatisch informiert und wird alle diejenigen Jobs anstoßen, die auf diese Bedingung gewartet haben.

Weitere Informationen siehe:

- *Nachrichten* im *Benutzerhandbuch*
- *Globale Nachrichten für Ereignisse* in der *Systemverwaltung*-Dokumentation

Das Konzept der Mailboxen

Die folgende Abbildung veranschaulicht das Konzept der Mailboxen am Beispiel des Netzwerks für die Lohnbuchhaltung:



So wird z.B. Benutzer 1, zum Beispiel eine Mitarbeiterin, die in der Datenerfassung ist, darüber informiert, dass die Bedingung 1 („Erfassen-OK“) nicht erfüllt ist. Sie kann dann die notwendigen Maßnahmen durchführen, indem sie die Daten bezüglich der Personalanwesenheit vervollständigt. Danach quittiert sie das in ihrer Mailbox, und die Verarbeitung wird fortgesetzt. Benutzer 3, der der Assistent des Personal-Chefs sein könnte, wird über jede nicht erfüllte Bedingung benachrichtigt und kann daher die Abarbeitung des gesamten Job-Netzwerkes überwachen. Benutzer 3 und 4 werden benachrichtigt, wenn das Netzwerk vollständig abgearbeitet ist und die Belege verteilt werden können.

Bis zu 10 Mailboxen können mit einer Benutzerkennung verknüpft werden.

Weitere Informationen:

- *Nachricht versenden* (bei Beendigung des Jobs)
- *Globale Nachrichten für Ereignisse*

- *Mailboxen* im *Benutzerhandbuch*.
- Feld **Mailboxen** im Abschnitt *Felder: Benutzer-Definition und Profil* in der *Systemverwaltung*

9 Ressourcen

Um die Durchführung von Jobs von der Verfügbarkeit von Ressourcen abhängig zu machen, können die Funktionen des Entire System Server benutzt werden.

Ressourcen müssen zuerst in der Systemverwaltung definiert werden, bevor sie innerhalb von Entire Operations als Vorbedingung von Jobs verwendet werden können. Ressourcen regeln den Verarbeitungsfluss und können in beliebiger Anzahl definiert werden. Sie verhindern, dass Jobs parallel laufen, wenn alle sonstigen Voraussetzungen (z.B. Zeitfenster oder logische Eingabebedingungen) erfüllt sind.

Weitere Informationen siehe ***Verwendung von Ressourcen*** in diesem Dokument sowie *Ressourcen* in der *Systemverwaltung*-Dokumentation.

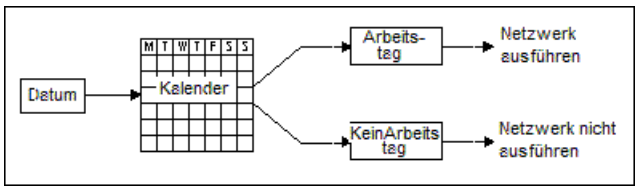
Im Beispiel des Netzwerkes aus der Lohnbuchhaltung (***Job-Netzwerk mit logischen Bedingungen***) kann verhindert werden, dass Job 6 (Personal-Report erstellen) und Job 7 (Gehaltsabrechnungen erstellen) parallel laufen. Hierzu kann eine Ressource definiert werden (z.B. CPU-Zeit), die in einer Anfangsmenge von 100 Einheiten vorhanden ist und von der jeder der beiden Jobs 60 Einheiten benötigt. Die zwei Jobs werden dann sequenziell abgearbeitet.

10 Kalender

Eine beliebige Anzahl von Kalendern kann auf sehr komfortable Art und Weise innerhalb des Systems definiert und verwaltet werden. Diese können dann entweder speziellen Eigentümern oder aber dem ganzen System zur Verfügung stehen.

Die Verwendung von Kalendern dient nur einem einzigen Zweck, nämlich zwischen Arbeitstagen und Feiertagen unterscheiden zu können. Entire Operations führt keine Aktivitäten an Feiertagen durch. Sie können festlegen, was mit einem Job-Netzwerk geschehen soll, das an einem solchen Tag eigentlich eingeplant war (Aktivierung vorher, Aktivierung nachher, ausfallen lassen).

Wenn Sie andererseits ein Job-Netzwerk z.B. jeden Freitag ausführen möchten brauchen Sie hierfür keinen Kalender.



Weitere Informationen siehe *Kalender* und *Kalender*-Verwaltung im *Benutzerhandbuch*.

11 Zeitpläne

Zeitpläne beinhalten die Datumsangaben für die geplante Ausführung von Job-Netzwerken. Sie können sowohl periodische als auch explizite Datumsangaben umfassen. Eine beliebige Anzahl von Zeitplänen ist definierbar, und ein Zeitplan kann von verschiedenen Job-Netzwerken referenziert werden.

Falls ein Zeitplan auf einem vordefinierten Kalender aufbaut, können Datumsangaben zur Ausführung von Job-Netzwerken sogar relativ zu Feiertagen gemacht werden (z.B. letzter Arbeitstag eines Monats).

Weitere Informationen siehe *Zeitpläne* und *Zeitplan-Verwaltung* im *Benutzerhandbuch*.

12 Symboltabellen und Symbole

Symboltabellen sind benutzerdefinierte Tabellen, die jeweils eine Liste von Symbolnamen mit ihren aktuellen Werten beinhalten. Diese Tabellen werden immer benutzt, wenn die auszuführenden Jobkontrollanweisungen zum Aktivierungs- oder Ausführungszeitpunkt erst aufgebaut werden sollen (dynamische Generierung von Jobkontrollanweisungen). Der Vorteil solcher Symboltabellen ist, dass sie nur einmal gepflegt werden müssen, aber in einer großen Anzahl von Jobs referenziert werden können.

Sie können eine beliebige Anzahl von Symboltabellen definieren und sie einfach dadurch benutzen, indem sie ihren Namen in der Definition der entsprechenden Job-Netzwerke angeben.

Bei der Definition von Symbolen kann spezifiziert werden, dass bei der Aktivierung der sie verwendenden Job-Netzwerke eine Aufforderung an eine Mailbox zur Dateneingabe erfolgen soll: die am Bildschirm eingegebenen Daten werden dann in die auszuführenden Jobkontrollanweisungen übernommen.

Bei jeder Netzwerk-Aktivierung wird eine Kopie der verknüpften Symboltabelle(n) erstellt. Damit wird es Ihnen möglich, Netzwerke mit unterschiedlichen Parametersätzen zur Ausführung vorzubereiten, und das sogar über einen längeren Zeitraum im voraus.

Wie können nun Symbole in den Jobkontrollanweisungen benutzt werden? Immer wenn ein Symbolname in den Jobkontrollanweisungen oder in einer Prozedur vorkommt, wird er durch seinen aktuellen Wert aus der Symboltabelle ersetzt. Den Zeitpunkt dieser Ersetzung können Sie selbst anhand zweier Steuerzeichen bestimmen: entweder den Zeitpunkt der Netzwerk-Aktivierung oder aber den Zeitpunkt der tatsächlichen Job-Ausführung.

Es gibt in Entire Operations auch eine große Anzahl vordefinierter (eingebauter) Symbole.

Die Symbole selbst werden in verschiedenen Tabellen (wie in Steplibs) gesucht. Symbole können rekursiv andere Symbole beinhalten; Systemvariablen können ebenfalls zur Bildung von Symbolwerten herangezogen werden.

Symboltabellen gehören Eigentümern. Ein Eigentümer kann eine beliebige Anzahl von Symboltabellen besitzen. Ein Benutzer kann alle Symboltabellen aller Eigentümer verändern, für die er autorisiert ist.

Symbole können mit Hilfe sogenannter APIs (Application Programming Interfaces) von beliebigen Natural Programmen abgefragt und modifiziert werden. Wird ein solches Natural Programm als Teil eines Entire Operations Job-Netzwerks ausgeführt, so können aktive Symboltabellen sogar während der laufenden Ausführung von Job-Netzwerken verändert werden.

Weitere Informationen siehe *Symboltabellen und Symbole* (in der *Systemübersicht*) und *Symboltabellen und Symbole* im *Benutzerhandbuch*.

13 Laufnummern

- Laufnummern 42
- Laufnummern-Bereich 42
- Reservierte Laufnummern 42

Weitere Informationen siehe *Laufnummern* im *Benutzerhandbuch*.

Laufnummern

Aktive Objekte in Entire Operations werden zusätzlich durch eine Laufnummer identifiziert, die ihnen automatisch bei der Erzeugung eines aktiven Objekts zugewiesen wird. Aktive Netzwerke oder Jobs werden bei einer Aktivierung eines Netzwerks oder eines Jobs erzeugt.

- Laufnummern sind auf Job-Netzwerk-Ebene eindeutig.
- Laufnummern können auch geplanten Aktivierungen zugewiesen werden. In der Planungsphase sind für eine gegebene Laufnummer keine aktiven Jobs vorhanden.
- Es gibt *keine Garantie* dafür, dass die Nummerierung der Netzwerk-Aktivierung mit den Aktivierungszeiten ansteigt.

Laufnummern-Bereich

Bei einem neuen Netzwerk beginnt die Erzeugung von Laufnummern mit 1. Wenn die höchstzulässige Laufnummer erreicht ist, beginnt die Nummerierung aktiver Netzwerke wieder mit 1. Importierte Job-Netzwerke fahren mit der Nummerierung aus der früheren Umgebung fort.

Die höchstzulässige Laufnummer kann bei den Entire Operations-Standardwerten definiert werden (Limit für Laufnummern).

Reservierte Laufnummern

Einige Laufnummern werden von Entire Operations zu speziellen Zwecken verwendet. Diese Nummern können nicht für Benutzer-Netzwerke und Jobs verwendet werden.

Tabelle der reservierten Laufnummern

Laufnummer	Äquivalent	Verwendung
-1	`abs`	Interne Darstellung der Bedingungs-Referenz `absolut`.
-2	`void`	Interne Darstellung der Bedingungs-Referenz `void` (leer).
-3	`K-RUN-MACROTEST`	Beim Testen von Makro-Programmen.
-4	`K-RUN-PREGENERATED`	Für die Speicherung vorgenerierter JCL.

14 Objekt-Versionierung

- Nutzung des Versionierungskonzepts 44
- Versionierung von Job-Netzwerken 44
- Versionierung von Symboltabellen 49

Entire Operations bietet Ihnen bei den Objekttypen **Job-Netzwerk** und **Symboltabelle** die Möglichkeit, mit verschiedenen Versionen eines Objekts zu arbeiten.

Dieser Abschnitt beschreibt das Versionierungskonzept. Weitere Informationen sind an den relevanten Stellen in den Beschreibungen der betreffenden Objekttypen im *Benutzerhandbuch* vorhanden.

Dieser Abschnitt behandelt folgende Themen:

Nutzung des Versionierungskonzepts

Die Versionierung von Job-Netzwerken und Symboltabellen ist optional.

Sie können bei jedem Job-Netzwerken und bei jeder Symboltabelle individuell entscheiden, ob Sie mit Versionen des betreffenden Objekts arbeiten wollen oder nicht.

Sie können die Versionierung in folgenden Fällen anwenden:

- Archivierung früherer Job-Netzwerk-Versionen, um diese zu einem späteren Zeitpunkt manuell aktivieren zu können.
- Archivierung früherer Symboltabellen-Versionen.
- Bereitstellung neuer Job-Netzwerk-Versionen oder Symboltabellen-Versionen für zukünftige Verwendungen.

Sie können Datumsbereiche für Netzwerk-Versions-Verwendung bzw. Datumsbereiche für Symboltabellen-Versions-Verwendung definieren, um festzulegen, welche Version im Falle von Zeitplan-Aktivierungen verwendet werden soll.

Versionierung von Job-Netzwerken

Dieser Abschnitt behandelt folgende Themen:

- Versionsnamen
- Versionsnamen-Exit
- Reservierte Versionsnamen für Netzwerke
- Erstellen von Netzwerk-Versionen durch Klonen
- Kopieren von Jobs
- Löschen von Netzwerk-Versionen
- Löschen von Netzwerk-Versionen oder einzelnen Jobs per API
- Verwendung der Netzwerk-Versionen für Zeitplan-Aktivierungen
- Berichte
- Import / Export
- Exit-Funktionalität

- Maximale Anzahl Versionen pro Netzwerk

Versionsnamen

Versionsnamen können maximal 10 Bytes lang sein. Die Namensvergabe ist bis auf wenige Einschränkungen beliebig.

Es gelten folgende Konventionen, Einschränkungen bzw. Empfehlungen:

- Groß-/ Kleinschreibung ist erlaubt.
- Leerzeichen und die Zeichen `?`, `<`, `>` sind in Versionsnamen nicht erlaubt.
- Wegen der reservierten Namen dürfen Versionsnamen nicht mit `,(,` anfangen.
- Wegen der Platzhalterzeichen-Behandlung („Wildcard“) dürfen Versionsnamen keinen Stern (*) enthalten.
- Um Probleme mit der Portierung zwischen verschiedenen Plattformen zu vermeiden, sollten Sonderzeichen und Umlaute vermieden werden.

Versionsnamen-Exit

Mit einem globalen Versionsnamen-Exit kann die Einhaltung einer kundenspezifischen Versionsnamen-Syntax erzwungen werden.

Weitere Informationen siehe *Globaler Exit für Versionsnamen* in der *Systemverwaltung*-Dokumentation.

Reservierte Versionsnamen für Netzwerke

<leer> ; in Selektionen und im Log auch: (unnamed)

Wird für die unbenannte Version verwendet.

Nach einer Migration von einer Entire Operations-Version vor 5.4.1 ist dies die einzige, immer vorhandene Netzwerk-Version.

In Parameterlisten (z. B. für die Berichte) kann man auch einen Bindestrich (-) angeben.

(current)

Wird durch die Version ersetzt, die für den gegebenen Tag als aktuelle Version für Zeitplan-Aktivierungen bestimmt ist.

(current) kann in Versions-Referenzen verwendet werden.

Erstellen von Netzwerk-Versionen durch Klonen

Die Kopier-Funktion für Job-Netzwerke wird auch für das Klonen von Job-Netzwerken und damit zur Erzeugung von Versionen verwendet.

Dies ist ein gängiger Weg zur Erstellung neuer Netzwerk-Versionen.

Man kann auch die Import-Funktion verwenden, um eine Version hinzuzufügen.

Kopieren von Jobs

Einzelne Jobs können aus einer beliebigen Version des Ursprungs-Netzwerks kopiert werden.

Löschen von Netzwerk-Versionen

Für das Löschen von Netzwerk-Versionen gilt Folgendes:

- Wenn mehrere Versionen eines Job-Netzwerks existieren, muss man eine der Versionen zum Löschen auswählen.
- Erst wenn die letzte (oder einzige) Version gelöscht wird, wird automatisch auch das „Network Main“-Objekt gelöscht.
- Versions-unabhängige Objekte, die zum Netzwerk gehören, werden erst mit dem „Network Main“-Objekt gelöscht.
- Eine Netzwerk-Version kann nicht gelöscht werden, wenn sie für mindestens einen aktuellen oder zukünftigen Datumsbereich als Standard-Version für Zeitplan-Aktivierungen definiert ist. Ein definierter Datumsbereich in der Vergangenheit ist für die Löschung einer Version unerheblich.

Löschen von Netzwerk-Versionen oder einzelnen Jobs per API

Mit der Anwendungsprogrammierungsschnittstelle `NOPUAC5N` (Funktion `D`, Laufnummer `-1`) können einzelne Netzwerk-Versionen sowie einzelne Jobs darin gelöscht werden.

Verwendung der Netzwerk-Versionen für Zeitplan-Aktivierungen

Zur Verwaltung der Versions-Verwendungen steht eine entsprechende Funktion zur Verfügung, die mit einem Kontextmenü-Kommando aufgerufen werden kann.

Weitere Informationen siehe *Datumsbereiche für Netzwerk-Versions-Verwendung verwalten* im *Benutzerhandbuch*.

Weitere Informationen siehe unten:

- Auswertung und Aktivierung der Netzwerk-Versionen
- Manuelle Aktivierung

- Aktivierung als Unternetzwerk
- Aktivierung als Job-Ende-Aktion
- Aktivierung mittels API
- Versionen ohne Zeitplan-Aktivierung
- Historie der Netzwerk-Aktivierungen – Tages-Ansicht

Auswertung und Aktivierung der Netzwerk-Versionen

Für die Auswertung der Datumsbereiche gilt Folgendes:

- Wenn das Netzwerk nur eine Version hat, wird immer diese Version aktiviert. Eine Zeitplanbereichs-Definition für die einzige Version wird ignoriert.
- Wenn das Netzwerk mehrere Versionen hat, so wird geprüft, ob eine dieser Versionen für den Aktivierungstag als aktuelle Version definiert ist. Mit anderen Worten: Es wird geprüft, ob der Aktivierungstag in einen der definierten Datumsbereiche fällt. Wenn dies zutrifft, so wird die für den zutreffenden Datumsbereich definierte Version aktiviert.
- Wenn für ein Netzwerk Datumsbereiche für die Netzwerk-Versions-Verwendung definiert sind, aber der Aktivierungstag in kein Intervall fällt, wird das Netzwerk nicht aktiviert, obwohl eine Zeitplan-Aktivierung vorgesehen ist. Entsprechende Protokolleinträge („Log-Meldungen“) und Benachrichtigungen werden ausgegeben.

Manuelle Aktivierung

Bei einer manuellen Aktivierung kann eine beliebige Netzwerk-Version gewählt werden. Die Standard-Version für Zeitplan-Aktivierungen des aktuellen Datumsbereichs (falls existent) wird zuerst angeboten.

Aktivierung als Unternetzwerk

In der Unternetzwerk-Definition können beliebige Versionen oder der reservierte Name `(current)` definiert werden.

Aktivierung als Job-Ende-Aktion

Für die Netzwerk- oder Job-Aktivierung als Job-Ende-Aktion können beliebige Versionen oder der reservierte Name `(current)` definiert werden.

Aktivierung mittels API

Für die Netzwerk- oder Job-Aktivierung mit der Anwendungsprogrammierungsschnittstelle `NOPUAC5N` können im Feld `NETWORK-VERSION` beliebige Versionen oder der reservierte Name `(current)` definiert werden.

Bitte beachten Sie, dass die API versionsbezogene Returncodes ausgeben kann.

Versionen ohne Zeitplan-Aktivierung

Man kann in Entire Operations beliebig viele Versionen eines Job-Netzwerks speichern. Versionen, die nicht (oder nicht mehr) in einem Verwendungsbereich für Zeitplan-Aktivierung definiert sind, werden nicht (mehr) automatisch aktiviert.

Historie der Netzwerk-Aktivierungen – Tages-Ansicht

Die Ausführungshistorie der Netzwerk-Starts enthält die Netzwerk-Version für jeden Lauf. Weitere Informationen siehe *Ausführungshistorie eines Netzwerks anzeigen* im *Benutzerhandbuch*.

Berichte

Netzwerk-Versionen werden bei der Generierung von Berichten berücksichtigt.

Import / Export

Netzwerk-Versionen werden berücksichtigt.

Exit-Funktionalität

Entire Operations-Exits, die mit Job-Netzwerken zu tun haben, unterstützen die Netzwerk-Versionierung.

Maximale Anzahl Versionen pro Netzwerk

Die maximale Anzahl von Netzwerk-Versionen kann in der Systemverwaltung systemweit eingeschränkt werden.

Weitere Informationen siehe Feld **Max. Anzahl Versionen pro Netzwerk oder Symboltabelle** auf der Registerkarte **Netzwerk-Optionen** im Abschnitt *Standardwerte für Netzwerk-Optionen* in der *Systemverwaltung*-Dokumentation.

Versionierung von Symboltabellen

Dieser Abschnitt behandelt folgende Themen:

- Versionsnamen
- Versionsnamen-Exit
- Reservierte Versionsnamen für Symboltabellen
- Erstellen von Symboltabellen-Versionen durch Klonen
- Kopieren von einzelnen Symbolen
- Löschen von Symboltabellen-Versionen
- Löschen von Symboltabellen-Versionen oder einzelnen Symbolen mittels API
- Gültigkeits-Zeiträume für Standard-Versionen
- Definition von Symboltabellen-Versionen
- Aktive Symboltabellen
- Symbolabfrage
- Reihenfolge bei der Symbolsuche
- Symboltabellen auf System- und Eigentümer-Ebene
- Protokollierung (Logging)
- Cross-Referenzen (XREF)
- Berichte
- Import / Export
- Exit-Funktionalität
- Anzahl Versionen pro Symboltabelle

Versionsnamen

Versionsnamen können maximal 10 Bytes lang sein. Die Namensvergabe ist bis auf wenige Einschränkungen beliebig.

Es gelten folgende Konventionen, Einschränkungen bzw. Empfehlungen:

- Groß- / Kleinschreibung ist erlaubt.
- Leerzeichen und die Zeichen `?`, `<`, `>` sind in Versionsnamen nicht erlaubt.
- Wegen der reservierten Namen dürfen Versionsnamen nicht mit `(` anfangen.
- Wegen der Platzhalterzeichen-Behandlung („Wildcard“) dürfen Versionsnamen keinen Stern (*) enthalten.
- Um Probleme mit der Portierung zwischen verschiedenen Plattformen zu vermeiden, sollten Sonderzeichen und Umlaute vermieden werden.

Versionsnamen-Exit

Mit einem globalen Versionsnamen-Exit kann die Einhaltung einer kundenspezifischen Versionsnamen-Syntax erzwungen werden.

Weitere Informationen siehe *Globaler Exit für Versionsnamen* in der *Systemverwaltung*-Dokumentation.

Reservierte Versionsnamen für Symboltabellen

Zusätzlich zu festen Versionsnamen gibt es einige reservierte Namen.

<leer> ; in Selektionen und im Log auch: (unnamed)

Wird für die unbenannte Version verwendet.

Nach einer Migration von einer Entire Operations-Version vor 5.4.1 ist dies die einzige, immer vorhandene Netzwerk-Version.

In Parameterlisten (z. B. für die Berichte) kann man auch einen Bindestrich (-) angeben.

- **(current)**
  Wird durch die Version ersetzt, die für den gegebenen Tag als aktuelle Version für Zeitplan-Aktivierungen bestimmt ist.

  `(current)` kann in Versions-Referenzen verwendet werden.
- **(nv)**
  Wird durch die Netzwerk-Version des verwendenden Netzwerks ersetzt.

  Falls es nur eine Netzwerk-Version ohne Namen gibt, so wird hier eine Symboltabellen-Version ohne Namen referenziert.

  Falls keine zur Netzwerk-Version namensgleiche Symboltabellen-Version vorhanden ist, wird die jeweilige Aktion mit Fehlermeldung abgebrochen.

  `(nv)` kann in Versions-Referenzen verwendet werden.
- **(svn)**
  Wird durch die Symboltabellen-Version des verwendenden Netzwerks ersetzt.

  `(svn)` kann in Versions-Referenzen verwendet werden, die einem Netzwerk untergeordnet sind.

  Anwendung u.a. für:

  - Job-Definition
  - alle Stellen, an denen man auch `(svj)` definieren kann.

- **(svj)**
  Wird durch die Symboltabellen-Version des verwendenden Jobs ersetzt.

  `(svj)` kann in Versions-Referenzen verwendet werden, die einem Job untergeordnet sind.

  Anwendung u.a. für:

  - Eingabebedingung abhängig von Symbolwert
  - Eingabebedingung abhängig von mehrfachem Symbol
  - Job-Ende-Aktion: Symbol setzen

Erstellen von Symboltabellen-Versionen durch Klonen

Die Kopier-Funktion für Symboltabellen wird auch für das Klonen von Job-Netzwerken und damit zur Erzeugung von Versionen verwendet.

Dies ist ein gängiger Weg zur Erstellung neuer Symboltabellen-Versionen.

Man kann auch die Import-Funktion verwenden, um eine Version hinzuzufügen.

Kopieren von einzelnen Symbolen

Einzelne Symbole können aus einer beliebigen Version der Ursprungs-Symboltabelle kopiert werden.

Löschen von Symboltabellen-Versionen

Für das Löschen von Symboltabellen-Versionen gilt Folgendes:

- Wenn mehrere Versionen einer Symboltabelle existieren, muss man eine der Versionen zum Löschen auswählen.
- Eine Symboltabellen-Version kann nicht gelöscht werden, wenn sie für mindestens einen aktuellen oder zukünftigen Datumsbereich als Standard-Version für Zeitplan-Aktivierungen definiert ist. Ein definierter Datumsbereich in der Vergangenheit ist für die Löschung einer Version unerheblich.

Löschen von Symboltabellen-Versionen oder einzelnen Symbolen mittels API

Mit dem Anwendungsprogrammierungsschnittstelle `NOPUSY6N` können einzelne Symboltabellen-Versionen sowie einzelne Symbole darin gelöscht werden.

Gültigkeits-Zeiträume für Standard-Versionen

Zur Verwaltung der Gültigkeits-Zeiträume steht eine Funktion zur Verfügung, die mit einem Kontextmenü-Kommando aufgerufen werden kann.

Weitere Informationen siehe *Datumsbereiche für Symboltabellen-Versions-Verwendung verwalten* im *Benutzerhandbuch*.

Definition von Symboltabellen-Versionen

Die Definition von Symboltabellen-Versionen ist möglich in der

- Netzwerk-Versions-Definition
- Job-Definition

Aktive Symboltabellen

Für aktive Symboltabellen gilt Folgendes:

- Die Aktivierung von Symboltabellen ist Bestandteil von Netzwerk- und Job-Aktivierungen.
- Eine Symboltabelle kann nur mit einer eindeutigen Version aktiviert werden. Die Bestimmung der zu verwendenden Symboltabellen-Version ist Bestandteil des Aktivierungsvorgangs.
- Aktive Symboltabellen haben nicht mehr die Versionsbezeichnungen `(current)` oder `(nv)`. Diese werden während der Aktivierung von Symboltabellen eindeutig aufgelöst.
- Aktive Symboltabellen können nur noch die Versionsbezeichnung `(none)` (= leer) oder einen festen Versionsnamen haben.
- Wenn eine benötigte Symboltabellen-Version fehlt, oder die Symboltabellen-Version nicht eindeutig bestimmt werden kann, ist ein Aktivierungsvorgang mit Fehlermeldung abzubrechen.
- Keinesfalls darf in einer nicht eindeutigen Situation eine Symboltabellen-Version „geraten“ werden.

Symbolabfrage

Die zu verwendende Symboltabellen-Version wird eindeutig bestimmt, bevor die Symboleingabe (bei manueller Aktvierung oder vor der Ausführung des Symbolabfrage-Exit im Monitor) erfolgt.

Reihenfolge bei der Symbolsuche

Die Reihenfolge, in der nach Symbolen in den in Ihrer Umgebung definierten Symboltabellen gesucht wird, ist abhängig von den hierarchischen Ebenen, auf denen auf die Symboltabellen in Ihrer Umgebung zugegriffen werden kann.

Symboltabellen auf System- und Eigentümer-Ebene

Symboltabellen auf Systemebene und Eigentümer-Ebene werden nicht versioniert. Es handelt sich um die Symboltabellen:

```
SYSDBA / A
<owner> / A
```

Protokollierung (Logging)

Das Protokoll aller Symbol-Aktionen enthält die Version der Tabelle, aus der das Symbol stammt.

Die von Entire Operations im JCL-Header generierten Kommentare enthalten für die verwendeten Symbole die Symboltabellen-Version.

Cross-Referenzen (XREF)

Symboltabellen-Versionen werden berücksichtigt.

Berichte

Symboltabellen-Versionen werden berücksichtigt.

Import / Export

Symboltabellen-Versionen werden berücksichtigt.

Exit-Funktionalität

Entire Operations-Exits, die mit Symbolen zu tun haben, unterstützen die Symboltabellen-Versionierung.

Beispiele:

- Symbolabfrage-Exit
- NOPUSY*x*N

Anzahl Versionen pro Symboltabelle

Die maximale Anzahl von Symboltabellen-Versionen kann in der Systemverwaltung systemweit eingeschränkt werden.

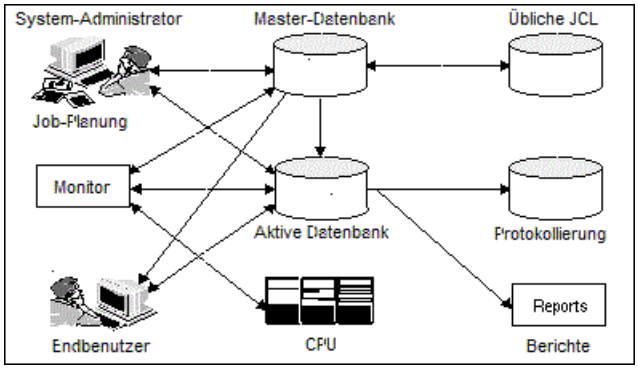
Weitere Informationen siehe Feld **Max. Anzahl Versionen pro Netzwerk oder Symboltabelle** auf der Registerkarte **Netzwerk-Optionen** im Abschnitt *Standardwerte für Netzwerk-Optionen* in der *Systemverwaltung*-Dokumentation.

III Entire Operations-Komponenten

15 Entire Operations-Komponenten

- Master-Datenbank 58
- Externe Jobkontrollanweisungen 59
- Aktive Datenbank 59
- Entire Operations-Monitor 61
- Betriebssystem 61
- Protokollierungsfunktionalität (Logging) 62
- Berichtsfunktionalität (Reporting) 62
- Editor 63

Entire Operations umfaßt folgende Komponenten:



Master-Datenbank

Alle Definitionen und Informationen, die Benutzer, Job-Netzwerke, Jobs und Zeitpläne betreffen, werden in der Master-Datenbank abgelegt. Die Master-Datenbank ist eine Adabas-Datei, was automatisch Benutzer-Synchronisation, Datenintegrität, Datenkomprimierung, automatische Dateierweiterung und automatische Wiederanlauffähigkeit bedeutet. Alle gespeicherten Objekte können am Bildschirm aus jeder der Host-Ablaufumgebungen Com-plete, CICS, IMS, TSO, TIAM und UTM sowie unter UNIX verwaltet werden. Zu diesen Objekten zählen:

- Benutzer-Profile
- Definitionen der Job-Netzwerke
- Job-Definitionen
- Eingabe- und Ausgabebedingungen
- Definitionen der Ressourcen
- Zeitpläne
- Kalender
- Symboltabellen

Externe Jobkontrollanweisungen

Mit Entire Operations können Jobkontrollanweisungen unverändert in Job-Netzwerke integriert werden, sie können sogar an ihrem ursprünglichen Speicherort bleiben. Hier werden sequentielle Dateien, PDS-, z/VSE- und LMS-Bibliotheken sowie die Speicherorte NATURAL und CA-LIBRARIAN unterstützt. Unter dem Betriebssystem UNIX können beliebige Shell-Prozeduren in die Steuerung von Entire Operations einbezogen werden.

Ein Kopieren der Jobkontrollanweisungen in die Master-Datenbank ist über die Import-Funktion möglich, und sollte immer dann benutzt werden, wenn besondere Sicherheitskriterien dies erfordern (Zugriff ist dann nur noch über Natural Security möglich) oder falls die Jobkontrollanweisungen mit der Master-Datenbank gesichert werden sollen.

Aktive Datenbank

Immer wenn ein Job-Netzwerk aktiviert wird, wird eine Kopie in die aktive Datenbank eingestellt. Ein Netzwerk wird hierbei entweder automatisch vom Entire Operations-Monitor entsprechend der Zeitplan-Daten oder aber durch einen Benutzer auf Anforderung aktiviert. Die aktive Datenbank kann daher auch mehrere Kopien desselben Job-Netzwerks beinhalten, jede von ihnen wird durch eine eindeutige Laufnummer identifiziert.

Folgende Informationen werden abgelegt:

- aktuelle Definition eingeplanter Job-Netzwerke zusammen mit ihren aktuellen Symboltabellen;
- aktive Jobkontrollanweisungen (dies bedeutet, dass alle Informationen über Jobkontrollanweisungen aus externen Speichermedien wie z.B. PDS, LMS, VSE-LIBRARIAN oder UNIX-Dateien in die aktive Datenbank kopiert werden);
- aktueller Status der Eingabe- und Ausgabebedingungen;
- aktueller Job-Status.

Sie können auf die aktive Datenbank auf die gleiche Art und Weise zugreifen und ihre Informationen modifizieren wie auf die Master-Datenbank. Änderungen an einem Objekt der aktiven Datenbank sind nur für den zugehörigen Netzwerklauf wirksam und haben keinen Einfluss auf die Definitionen der Netzwerke und Jobs in der Master-Datenbank. Hiermit wird es Ihnen möglich, Änderungen vorzunehmen, die nur an ganz bestimmten Produktionstagen Gültigkeit haben sollen.

Grundsätzliches zu Aktiven Netzwerken und Jobs in der aktiven Datenbank

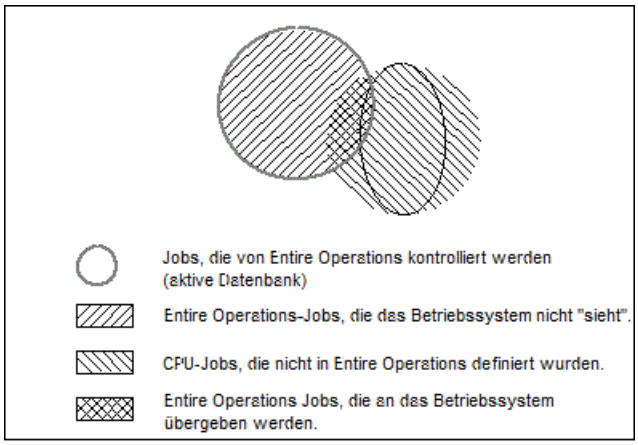
Wenn Entire Operations ein Job-Netzwerk aktiviert, wird in der aktiven Datenbank eine Kopie des in der Master-Datenbank definierten Netzwerks gezogen, und es wird ihr eine Laufnummer zugeordnet. Es können mehrere Kopien desselben Netzwerks in der aktiven Datenbank vorhanden sein. Diese Kopien unterscheiden sich durch ihre Laufnummern.

Zu diesem Zeitpunkt stehen alle Jobs ihren Abhängigkeiten entsprechend zur Ausführung bereit. Sie stehen *in der aktiven Datenbank*. Es werden aber nicht alle Jobs in der aktiven Datenbank zu Betriebssystem-Jobs. Zum Beispiel sind Jobs vom Typ „Dummy“ oder Natural-Programme Entire Operations-Jobs, die nicht dem Betriebssystem übergeben werden.

Es wird also zwischen zwei Gruppen von Jobs im Rechenzentrum unterschieden:

- **Jobs in der aktiven Datenbank von Entire Operations**
  (einschließlich nicht dem Betriebssystem übergebener Jobs);
- **Betriebssystem-Jobs**
  (einschließlich Entire Operations nicht bekanntgegebener Jobs).

Die folgende graphische Darstellung zeigt diese zwei Gruppen:



Der Kreis oben links enthält alle Jobs, die von Entire Operations kontrolliert werden. Entire Operations kann keine Jobs steuern, die nicht in Entire Operations definiert worden sind.

Jobs außerhalb von Entire Operations laufen unbeeinträchtigt und *unbemerkt* vom Entire Operations-Monitor auf dem Computer. Andererseits hält Entire Operations Informationen über Jobs bereit, die *unbemerkt* vom Betriebssystem laufen.

Die aktive Datenbank befindet sich in der aktiven Entire Operations-Datenbank und enthält alle betriebsrelevanten Informationen des aktuellen Laufes der aktivierten Job-Netzwerke.

Sie können auf die aktive Datenbank zugreifen, um aktive Netzwerke und Jobs zu pflegen, einschließlich logischer Bedingungen, Ressourcen und Zeitplan-Parameter.

Entire Operations-Monitor

Der Monitor ist das Herz von Entire Operations. Er ist ein komplexes Programm, das periodisch aktiviert wird und die Definitionen der Master-Datenbank daraufhin überprüft, ob eine Verarbeitung ansteht. Der Monitor aktiviert Netzwerke und bringt Jobs unter Berücksichtigung ihrer Vorbedingungen zur Ausführung; er kontrolliert die Abarbeitung von Job-Netzwerken, auch dann, wenn sie auf verschiedenen Rechnern stattfindet.

Der Monitor führt die folgenden Funktionen durch:

- er aktiviert automatisch eingeplante Netzwerke (und kopiert sie dabei in die aktive Datenbank);
- er überprüft Zeitfenster für die Ausführung von Jobs und Job-Netzwerken;
- er überprüft Eingabebedingungen und Ressourcen;
- er bringt Jobs unter Berücksichtigung ihrer (internen) Prioritäten zur Ausführung;
- er überwacht Jobs in den verschiedenen Warteschlangen des Betriebssystems;
- er analysiert den Job-Ende-Status, stellt das Eintreten von Ereignissen fest und stößt entsprechende Systemaktionen (Bedingungen setzen, Meldungen schicken, Programme starten) an;
- er protokolliert alle wichtigen Informationen.

Es ist möglich, die Funktionen des Entire Operations-Monitors auf mehrere Unterfunktionen (Subtasks) zu verteilen. Durch dieses Subtasking können Verarbeitungsprozesse parallelisiert und so Multiprozessor-Architekturen zur Performance-Verbesserung ausgenutzt werden. Die Verteilung der typischen Monitorfunktionen nimmt der Systemadministrator vor.

Weitere Informationen siehe *Monitor-Standardwerte* in der *Systemverwaltung*-Dokumentation.

Betriebssystem

CPU-basierte Verarbeitung wird vom Entire Operations-Monitor auf den zugrundeliegenden Betriebssystemen initiiert: Jobs, Tasks, Prozeduren oder Natural-Programme werden gestartet.

Der Monitor schickt hierzu entsprechende Anforderungen an den ausführenden Entire System Server-Knoten, eventuell über Entire Net-Work auch netzwerkweit.

Protokollierungsfunktionalität (Logging)

Entire Operations bietet eine Protokollierungsfunktionalität an, mit deren Hilfe jedes Ereignis während einer Netzwerkdurchführung in einem Protokoll („Log“) aufgezeichnet werden kann. Es protokolliert darüber hinaus alle manuellen Veränderungen an den aktiven Jobs, wie z.B. Änderungen an Jobkontrollanweisungen oder Modifikationen an Symboltabellen. Diese Informationen sind dann online verfügbar und können für Berichte und statistische Zwecke genutzt werden.

Die protokollierten Informationen können nach verschiedenen Selektionskriterien ausgewertet werden: nach Job-Netzwerken und Jobs, nach Datums- und Zeitbereichen und nach Benutzern. Sie können auch in eine sequenzielle Datei entladen werden, um von dort aus nach frei wählbaren Kriterien mit beliebigen Werkzeugen weiterverarbeitet zu werden.

Die Aufzeichnung der Jobkontrollanweisungen selbst, des Job-Protokolls und/oder der mit dem Job verknüpfter Meldungen kann entweder für alle oder aber ausgewählte Jobs angefordert werden.

Berichtsfunktionalität (Reporting)

Die Berichtsfunktionalität („Reporting“) stellt eine große Auswahl an Berichtsarten zur Verfügung, die insbesondere für die Verfolgung aller durchgeführten Aktivitäten sowie als Grundlage für Zeitplanungen oder sogar für zukünftige Produktionszyklen nützlich sein können.

Berichte bauen auf vorhandenen Protokollinformationen („Log“) auf. Sie können für alle Jobs oder aber für diejenigen, die innerhalb eines Datums-/Zeitfensters ausgeführt wurden angefordert werden, wobei diese Auswahl auf alle beendeten oder aber alle abgebrochenen Jobs eingeschränkt werden kann.

Beschreibungen zu Job-Netzwerken stehen als kurze Übersicht oder aber in ausführlicher Form zur Verfügung.

Die Liste aller Jobs für ein bestimmtes Datum kann angezeigt werden. Dies macht die Vorhersage eines beliebigen zukünftigen Produktionstages möglich.

Alle Berichte können entweder online angezeigt oder aber zu Archivierungszwecken ausgedruckt werden.

Weitere Informationen siehe *Berichte* im *Benutzerhandbuch*.

Editor

Entire Operations stellt einen integrierten Editor zur Verfügung.

Sie können mit dem Editor folgende Funktionen ausführen:

- Jobkontrollanweisungen in der Master-Datenbank, entweder in einer externen Datenhaltung oder aber in einer Natural-Datei;
- Jobkontrollanweisungen in der aktiven Datenbank;
- Natural Programme und User Exits;
- Online Dokumentation von Netzwerken, Jobs und beliebigen Ereignissen (z.B. als Notizzettel-Information);
- Job-Protokolle und Job-Ausgabelisten (nur im Anzeigemodus).

IV Entire Operations-Funktionen

16 Entire Operations-Funktionen

- Entire Systems Management-Hauptbildschirm 68
- Verwaltung der Job-Netzwerke 68
- Zeitplanung für Job-Netzwerke 68
- Job-Verwaltung 69
- Job-Zeitplanung 69
- Kalender-Definition 70
- Verwaltung der Bedingungen 70
- Job-Ende-Prüfung und -Aktionen 71
- Verwendung von Ressourcen 73
- Dynamische Generierung von Jobkontrollanweisungen (JCL) 73
- Editor für System-Objekte 78
- Berichtsfunktionen 79
- Cross-Referenzen 80
- Benachrichtigung 80

Entire Operations bietet eine große Anzahl Funktionen, die dem Benutzer die Definition von Objekten innerhalb des Systems und die Kontrolle und Überwachung der Ausführung von Netzwerken ermöglichen.

Entire Systems Management-Hauptbildschirm

Wenn Sie den Entire Operations GUI Client starten, erscheint der Entire Systems Management-Hauptbildschirm. Sie können von hier aus einer grafischen Benutzungoberfläche alle Entire Operations-Funktionen auf dem Großrechner aufrufen.

Weitere Informationen siehe *Elemente des Entire Systems Management-Hauptbildschirms* im *Benutzerhandbuch*.

Verwaltung der Job-Netzwerke

Bevor Entire Operations ein Job-Netzwerk identifizieren und aktivieren kann, muss dieses im System definiert werden. Zur Erläuterung kann eine kurze Beschreibung hinzugefügt werden. Eine Knoten-Nummer sollte als Standardwert für die Ausführung aller Jobs in diesem Netzwerk angegeben werden. Eine Symboltabelle kann ebenfalls angegeben werden, um so die Möglichkeit der dynamischen Generierung von Jobkontrollanweisungen zu nutzen. Die Knoten-Nummer eines Rechners wird als Standardwert für die Ausführung aller Jobs des Netzwerks herangezogen und kann für jeden Job überschrieben werden.

Weitere Informationen siehe *Netzwerk-Verwaltung* im *Benutzerhandbuch*.

Zeitplanung für Job-Netzwerke

Zeitplanung für ein Job-Netzwerk bedeutet, Datums- und Zeitangaben festzulegen, an denen dieses Netzwerk durchgeführt werden soll. Ein Job-Netzwerk wird aktiviert (d.h. in die aktiven Datenbank kopiert) entsprechend seiner eingeplanten Startzeit und unter Berücksichtigung weiterer Plan-Parameter des Netzwerks.

Während der Durchführung eines Job-Netzwerks überprüft Entire Operations, ob alle Vorbedingungen für einen Job erfüllt sind: Zeitfenster, Ressourcen und Eingabebedingungen. Entsprechend dieser Überprüfung wird der Job automatisch gestartet und seine Ausführung überwacht.

Weitere Informationen siehe *Zeitpläne* und *Zeitplan-Definition anlegen* im *Benutzerhandbuch*.

Job-Verwaltung

Innerhalb der Job-Netzwerk-Verwaltung kann der Benutzer die Liste aller zu einem Netzwerk gehörenden Jobs anfordern. Er kann dann Jobs hinzufügen, löschen oder aber verändern. Ein Job wird hierbei definiert durch einen (logischen) Namen, einen Typ und seinen Speicherort (PDS-Datei, Natural-Bibliothek usw.) Eine kurze Erläuterung des Jobs kann hinzugefügt werden. Daneben kann für die Ausführung dieses Jobs eine Knoten-Nummer angegeben werden, die dann den Standardwert überschreibt, der bei der Netzwerk-Definition angegeben wurde.

Wenn im Job die Möglichkeit der dynamischen Generierung von Jobkontrollanweisungen genutzt werden soll, muss die zu verwendende Symboltabelle ebenfalls angegeben werden (siehe hierzu auch den Abschnitt ***Dynamische Generierung von Jobkontrollanweisungen***). Der Benutzer kann dann alle derart referenzierten Jobkontrollanweisungen einfach über eine zugeordnete PF-Taste editieren.

Weitere Informationen siehe *Job-Verwaltung* im *Benutzerhandbuch*.

Job-Zeitplanung

Die Aktivierung des einzelnen Jobs hängt von seinen Zeitplanungsparametern ab. Genauso wie für ganze Netzwerke können auch für einzelne Jobs (früheste) Startzeiten vergeben werden.

Daneben kann der Benutzer angeben, ob ein eingeplanter Job, der aus irgendeinem Grund nicht durchgeführt wurde (z.B. wegen Hardware-Problemen) nochmals eingeplant werden soll: der Job würde dann zum nächsten Planungsdatum zweimal ausgeführt werden. Es kann darüber hinaus bestimmt werden, wie lange Jobs in der aktiven Datenbank auf ihre Ausführung warten dürfen. Auch kann der Benutzer dem Job eine geschätzte Laufzeit zuordnen: der Entire Operations-Monitor benutzt diese dann zur Berechnung der voraussichtlichen Ende-Zeit des Jobs.

Mit Hilfe der Zeitplan-Angaben eines Jobs ist es Entire Operations möglich, ausgewählte Benutzer des Systems davon zu unterrichten, dass der Start eines Jobs nicht erfolgt ist, um eine vorher definierte späteste Ende-Zeit („Deadline“) nicht zu überschreiten.

Kalender-Definition

Kalender werden in Zeitplänen referenziert, welche ihrerseits in der Netzwerk-Verwaltung definiert sind. Die Anzahl von Kalendern innerhalb des Systems ist beliebig. Kalender können einem Eigentümer gehören oder aber innerhalb des gesamten Systems zur Verfügung stehen. In der Kalender-Verwaltung kann der Benutzer diese hinzufügen, löschen oder verändern (systemweite Kalender können nur vom Administrator verändert werden).

Ein Kalender wird durch seinen Namen und eine Jahresangabe definiert. Die Definition neuer oder die Verwaltung bestehender Kalender besteht dann einfach darin, Feiertage zu markieren.

Entire Operations berücksichtigt diese Feiertage dann dadurch, dass es Netzwerke, die für diese Tage eingeplant wurden, nicht aktiviert. Weitergehende Regelungen für diesen Fall können getroffen werden, wie z.B. aktiviere Netzwerk am folgenden Werktag.

Weitere Informationen siehe *Kalender* im *Benutzerhandbuch*.

Verwaltung der Bedingungen

Logische Bedingungen stellen Abhängigkeiten zwischen Jobs dar. Sie werden mit einer dafür vorgesehenen Verwaltungsfunktion definiert, gelöscht oder aber verändert. Mit einem Job kann eine beliebige Anzahl logischer Bedingungen verknüpft werden. Eine logische Bedingung kann zwei Zustände annehmen, die dann die weitere Verarbeitung innerhalb von Entire Operations bestimmen: wahr (Bedingung existiert) oder falsch (Bedingung existiert nicht).

Jede Bedingung wird durch einen Namen und ein Referenzdatum identifiziert. Dies erlaubt es dem Entire Operations-Monitor, zwischen gleichen Ereignissen zu unterscheiden, die zu verschiedenen Zeitpunkten eingetreten sind. Zeitangaben können entweder in relativen oder absoluten Einheiten gemacht werden. Alle relativen Angaben werden in absolute umgerechnet, wenn der Job in die aktive Datenbank eingestellt wird.

Logische Bedingungen können verwendet werden als:

- **Eingabebedingungen**
  Sämtliche Eingabebedingungen müssen erfüllt sein, bevor Entire Operations einen zuvor aktivierten Job zur Ausführung bringen kann. Um nun zwei Jobs zu verknüpfen, muss eine Eingabebedingung des einen Jobs mit als Ausgabebedingung seines Vorgängers definiert werden. Eine Eingabebedingung kann durch ein CPU-basiertes Ereignis oder durch einen manuellen Benutzereingriff erfüllt werden.

  Neben dem Namen und dem Referenzdatum kann der Benutzer auch eine Mailbox mit einer Bedingung verknüpfen. Entire Operations wird dann automatisch alle Benutzer dieser Mailbox über alle ihre zugeordneten, ausstehenden Bedingungen informieren. Jedem Benutzer des Systems

können bis zu 10 Mailboxen zugeordnet werden: er wird so die Liste aller Meldungen sehen können, die an eine dieser Mailboxen geschickt worden ist

Daneben kann der Benutzer festlegen, in welchem Status eine Bedingung sein muss, damit ein Job zur Ausführung gebracht werden kann (wahr oder falsch), ob ein Job warten soll, bis ihm diese Bedingung exklusiv zur Verfügung steht (dies kann benutzt werden, um die parallele Ausführung von Jobs zu verhindern, die die gleiche Bedingung benutzen) und ob Entire Operations am Job-Ende die Bedingung automatisch zurücksetzen soll.

Vor der Durchführung eines Jobs überprüft der Entire Operations-Monitor automatisch alle Eingabebedingungen. Daneben können sie diese Prüfung auch von einer Natural-Benutzerroutine durchführen lassen: diese muss dazu bei der Definition der logischen Bedingung angegeben werden.

Weitere Informationen siehe *Eingabebedingungen für Job verwalten* im *Benutzerhandbuch*.

- **Ausgabebedingungen**
  Ausgabebedingungen werden vom Entire Operations-Monitor automatisch gepflegt, wenn das ihnen zugeordnete Ereignis eingetreten ist. In diesem Fall werden alle Jobs gestartet, für die diese Bedingung einzige Eingabebedingung ist. Ereignisse und Ausgabebedingungen werden in Entire Operations innerhalb der Job-Ende-Prüfungen festgelegt (siehe ***Jobende-Prüfung und -Aktionen***).

  Genau wie im Fall der Eingabebedingungen werden auch Ausgabebedingungen durch Namen und Referenzdatum definiert. Darüber hinaus kann der Benutzer bestimmen, ob die Ausgabebedingung bei Eintreffen des zugeordneten Ereignisses (auf wahr) gesetzt oder (auf falsch) zurückgesetzt werden soll.

  Bis zu 20 Ausgabebedingungen können mit einem einzelnen Ereignis verknüpft werden.

Job-Ende-Prüfung und -Aktionen

Die Job-Ende-Prüfung bezeichnet die Verarbeitung, mit der Entire Operations das Job-Ende feststellt.

Unmittelbar nach Job-Ende untersucht Entire Operations, ob benutzerdefinierte Ereignisse eingetreten sind. Solche Ereignisse können sein:

- ein Return-Code wurde in einem bestimmten Arbeitsschritt innerhalb eines Jobs gesetzt,
- ein Return-Code wurde in einem beliebigen Arbeitsschritt eines Jobs gesetzt,
- eine Zeichenkette wurde im Protokoll oder in den Ausgabe-Listen des Jobs gefunden,
- ein Natural-User Exit wurde ausgeführt, welcher seinerseits den Job-Ende-Status durch das Setzen eines Return-Codes anzeigt.

Diese Benutzerroutine kann:

- das Job-Protokoll oder die Ausgabelisten selbst untersuchen,
- alle vom Job erzeugten Daten lesen,
- Systemfunktionen durchführen,
- Meldungen verschicken.

Standard-Prüfungen

Je nach Betriebssystem, auf dem der Job durchgeführt wurde, unternimmt Entire Operations eine Reihe von Standard-Prüfungen zur Feststellung des Job-Ergebnisses. Für z/OS-Systeme werden z.B. Systemabbrüche oder Syntaxfehler in den Jobkontrollanweisungen automatisch erkannt. Diese Standard-Prüfungen werden für jeden Job durchgeführt, unabhängig davon, ob daneben noch benutzerdefinierte Prüfungen für diesen Job angefordert sind.

Erneute Versuche bei Job-Ende-Prüfung

Für das Betriebssystem z/OS gilt Folgendes:

Bei unvollständigem SYSOUT wird das Auslesen von SYSOUT zehnmal im Abstand von mindestens 30 Sekunden wiederholt. Der Abstand kann bei längerer Wartezeit des Monitor-Task länger sein.

Job-Ende-Aktionen

Für jedes definierte Ereignis kann der Benutzer festlegen, wie Entire Operations darauf reagieren soll. Folgende Möglichkeiten stehen zur Verfügung:

- verknüpfte Ausgabebedingungen automatisch setzen oder zurücksetzen (siehe *Ausgabebedingungen*);
- eine Nachricht schicken, und zwar wahlweise an:
  - einen bestimmten Betriebssystem-Benutzer,
  - die System-Konsole,
  - eine Entire Operations Mailbox,
  - an einen Benutzer des Software-AG-Bürokommunikationssystems Con-nect,
  - an eine E-Mail-Adresse.
- Job-Protokoll und Ausgabelisten löschen oder drucken;
- Fehlerbehebungsmaßnahmen einleiten (im Falle eines Job-Abbruchs).
- Datei-Übergabe an Entire Output Management.

Weitere Informationen siehe *Job-Ende-Prüfungen und -Aktionen* im *Benutzerhandbuch*.

Verwendung von Ressourcen

Ressourcen können entweder real existierende Ressourcen darstellen oder aber fiktiv sein.

Die Existenz realer Ressourcen kann mit Hilfe von Entire System Server-Funktionalität festgestellt werden. Zum Beispiel können Sie den auf jeder verfügbaren Platte vorhandenen Freiplatz bestimmen, das Vorhandensein einer beliebigen katalogisierten Datei oder die aktuelle Anzahl gerade laufender Jobs.

Dem gegenüber stehen die Ressourcen, die nur innerhalb von Entire Operations eine Bedeutung besitzen. Der System-Administrator kann für sie einen Anfangswert festlegen, der Benutzer kann dann diese Ressource zur Regulierung des Job-Flusses benutzen.

Definiert er zum Beispiel eine bestimmte Menge einer Ressource als Voraussetzung eines Jobs, wird dieser Job solange nicht ausgeführt, bis die geforderte Menge zur Verfügung steht. Der Benutzer kann also eine Kombination aus Ressourcen benutzen, um die gewünschte Reihenfolge der Job-Durchführung innerhalb eines Job-Netzwerks zu erreichen oder um die parallele Ausführung von Jobs zu verhindern.

Weitere Informationen siehe *Ressourcen* in der *Systemverwaltung*-Dokumentation.

Dynamische Generierung von Jobkontrollanweisungen (JCL)

Bei der Definition eines Jobs innerhalb eines Job-Netzwerks kann der Benutzer bestimmen, dass dessen Jobkontrollanweisungen dynamisch generiert werden, und zwar entweder zur Zeit der Jobaktivierung oder des Jobstarts.

Die dynamische Generierung von Jobkontrollanweisungen ist mit Hilfe der Entire Operations MACRO-Funktionalität realisiert, als Erweiterung der Programmiersprache Natural. Ein solches MACRO besteht aus normalen Natural-Anweisungen sowie Jobkontrollanweisungen beliebigen Formats. Innerhalb der Jobkontrollanweisungen können durch Steuerzeichen gekennzeichnete Variablen verwendet werden, die während der dynamischen Generierung durch ihre Werte ersetzt werden.

Diese aktuellen Werte werden aus den Symboltabellen entnommen, die die aktuellen, als Ersetzungswerte zu verwendenden Werte enthalten müssen. Die zu benutzenden Symboltabellenhierarchie kann im Dialog **Verwendbare Symboltabellen** bestimmt werden (siehe *Jobs und Netzwerke finden, die eine Symboltabelle verwenden* im *Benutzerhandbuch*. Weitere Informationen siehe *Fluchtzeichen definieren* im *Benutzerhandbuch*.

Zum Zeitpunkt der Variablenersetzung (entweder Aktivierung oder Ausführung) wird jedes in den Jobkontrollanweisungen referenzierte Symbol, das nicht in der Symboltabelle des Jobs gefunden wird, in der oder den Symboltabellen des Eigentümers SYSDBA gesucht. Die Anzahl

der Einträge, die ein Benutzer in einer Symboltabelle definieren kann, ist beliebig, die Anzahl der Symboltabellen ebenfalls.

Darüber hinaus stellt Entire Operations eine Reihe von Standardvariablen dem dynamisch zu generierenden Programm in einem Parameterabschnitt zur Verfügung, wie z.B. den Namen des Job-Eigentümers, des Jobs, des Job-Netzwerks und des ursprünglichen Zeitplan-Datums. Ebenso sind Natural-Systemvariablen z.B. für Datum (`*DAT*`), Zeit (`*TIM*`), Benutzerkennung (`*USER`) verfügbar. Da es möglich ist, all diese Parameter an beliebigen Stellen der Jobkontrollanweisungen einzubauen, können unterschiedliche Folgen der Jobkontrollanweisungen in Abhängigkeit von `*DAT*`, `*TIM*` usw. generiert werden.

Die Möglichkeit zur dynamischen Generierung von Jobkontrollanweisungen bietet Entire Operations auf allen unterstützen Plattformen an (z/OS, z/VSE, BS2000, UNIX).

Weitere Informationen siehe *Dynamische JCL-Generierung (JCL-Speicherart MAC)*.

Beispiele

- ***Beispiel 1: Dynamische Jobkontrollanweisungen in einer z/OS-Umgebung***
- ***Beispiel 2: Dynamische Jobkontrollanweisungen in einer BS2000-Umgebung***
- ***Beispiel 3: Dynamische Jobkontrollanweisungen in einer UNIX-Umgebung***

Beispiel 1: Dynamische Jobkontrollanweisungen in einer z/OS-Umgebung:

Die Symboltabelle des MACRO-Programms soll wie folgt aussehen:

Symbolname	Aktueller Wert
BIBLIOTHEK	SN.SYSF.SOURCE
KLASSE	G

Eine der Variablen im übergebenen Parameterabschnitt soll den Wert haben:

P-OWNER	NET1

Systemvariablen sollen die folgenden Werte besitzen:

*TPSYS	COMPLETE
*DEVICE	BATCH
*INIT-USER	SN

Die folgende Auflistung stellt nun ein Natural MACRO-Programm dar, das einen Parameterabschnitt und Jobkontrollanweisungen umfaßt. Variablennamen sind durch #gekennzeichnet und werden aus obiger Symboltabelle entnommen:

```
# DEFINE DATA PARAMETER USING NOPXPL-A
# LOCAL /* MUST BE CODED
# END-DEFINE
//SNMAC4 JOB ,#P-OWNER,MSGCLASS=X,CLASS=#CLASS //STEP01 EXEC
PGM=NOPCONTI,PARM='C0004' //STEPLIB  DD DISP=SHR,DSN=#STEPLIB
//* DEVICE: *DEVICE, INIT-USER: *INIT-USER //* TPSYS: *TPSYS
# IF CLASS = 'G'
//* THE MSGCLASS IS REALLY 'G'
# ELSE
//* ANOTHER MSG-CLASS FOUND
# END-IF
//*
```

Die hieraus dynamisch generierten Jobkontrollanweisungen lauten:

```
//SNMAC4 JOB ,NET1,MSGCLASS=X,CLASS=G
//STEP01 EXEC PGM=NOPCONTI,PARM='C0004' //STEPLIB  DD
DISP=SHR,DSN=SN.SYSF.SOURCE //* DEVICE: BATCH, INIT-USER: SN
//* TPSYS: COMPLETE
//* THE MSGCLASS IS REALLY 'G'
//*
```

Beispiel 2: Dynamische Jobkontrollanweisungen in einer BS2000-Umgebung:

Die Felder des View DB-INFO sollen hierbei nach dem Datenbank-Zugriff (FIND-Anweisung) folgende Werte haben:

Feld	Wert
NUCLEUS	055
LP1	1000
NU1	100
ACCOUNT	EXAMPLE
NH1	4000
MSG	FHL
VERSION	524

Die Variablen aus dem Parameterabschnitt sollen folgende Werte haben:

Variable	Wert
P-OWNER	OS
P-JOB	NUC055
P-EXECUTION-NODE	055

Es wird keine Symboltabelle für den MACRO-Job referenziert, alle Variablen sind durch ein vorangestelltes Steuerzeichen (hier: #) gekennzeichnet:

```
# DEFINE DATA PARAMETER USING NOPXPL-A
#  1 L-JOB
# 1 REDEFINE L-JOB
# 2  L-JOB-A    (A3)
# 2  L-JOB-NUC  (N3)
# LOCAL   /* LOCAL VARIABLES START HERE
# 1 DB-INFO VIEW OF DB-INFO
# 2  NUCLEUS
# 2  LP1
# 2  NU1
# 2  ACCOUNT
# 2  NH1
# 2  MSG
# 2  VERSION    /* E.G. 524
# 1 LWP  (N7)
# 1 NUC  (N3)
# 1 SPOOL  (A10) INIT <'NOSPOOL'>
# END-DEFINE
# *
# MOVE P-JOB TO L-JOB-A
# MOVE P-EXECUTION-NODE TO NUC
# F1. FIND DB-INFO WITH NUCLEUS = NUC
/.NUC NUC LOGON #P-OWNER,#ACCOUNT
/OPTION MSG=#MSG
/REMARK
/REMARK  NUCLEUS #NUC
/REMARK
/SYSFILE  SYSLST = NUC NUC..LST.NUC
/SYSFILE  SYSDTA = SYSCMD
/FILE  ADA VERSION..MOD,LINK=DDLIB
/FILE  *DUMMY,LINK=DDLOG
/FILE  *DUMMY,LINK=DDSIBA
/FILE  ADA NUC..ASSO,LINK=DDASSOR1,SHARUPD=YES
/FILE  ADA NUC..DATA,LINK=DDDATAR1,SHARUPD=YES
/FILE  ADA NUC..WORK,LINK=DDWORKR1,SHARUPD=YES
/EXEC  (ADARUN,ADA VERSION..MOD)
# COMPUTE LWP = F1.LP1 * (F1.NU1 + 100)
ADARUN PROG=ADANUC,LP=F1.LP1,LU=65535,LWP=#LWP ADARUN
DB=#NUC,NU=#NU1,NC=20,TT=600,TNAE=1800 ADARUN NH= NH1
/SYSFILE  SYSLST = (PRIMARY)
/SYSFILE  SYSDTA = (PRIMARY)
/SYSFILE  SYSOUT = (PRIMARY)
/LOGOFF  SPOOL
# END-FIND
```

Hieraus resultieren folgende dynamisch generierte Jobkontrollanweisungen:

```
/.NUC055 LOGON OS,EXAMPLE
/OPTION MSG=FHL
/REMARK
/REMARK  NUCLEUS 055
/REMARK
/SYSFILE  SYSLST = NUC055.LST.NUC
/SYSFILE  SYSDTA = SYSCMD
/FILE ADA524.MOD,LINK=DDLIB
/FILE *DUMMY,LINK=DDLOG
/FILE *DUMMY,LINK=DDSIBA
/FILE ADA055.ASSO,LINK=DDASSOR1,SHARUPD=YES
/FILE ADA055.DATA,LINK=DDDATAR1,SHARUPD=YES
/FILE ADA055.WORK,LINK=DDWORKR1,SHARUPD=YES
/EXEC (ADARUN,ADA524.MOD)
ADARUN PROG=ADANUC,LP=1000,LU=65535,LWP=200000 ADARUN
DB=055,NU=100,NC=20,TT=600,TNAE=1800 ADARUN NH=4000
/SYSFILE  SYSLST = (PRIMARY)
/SYSFILE  SYSDTA = (PRIMARY)
/SYSFILE  SYSOUT = (PRIMARY)
/LOGOFF NOSPOOL
```

Anmerkung: Jede JCL, die zur Aktivierungs-Zeit generiert wurde und die MACRO-Sprache benutzt, kann vom Benutzer geändert werden, bis der Job endgültig bestätigt wurde. Natürlich ist diese Veränderung nur für den aktuellen Netzwerk-Lauf gültig.

Beispiel 3: Dynamische Jobkontrollanweisungen in einer UNIX-Umgebung:

Im folgenden Beispiel soll eine dynamische Symbol-Ersetzung innerhalb eines Bourne-Shell-Skripts erfolgen (als Steuerzeichen dient hier §):

```
#
# Bourne shell script for checking the number of users
# entered in /etc/passwd.
# If more than §USER-LIMIT entries appear,
# the script will be ended with exit 1.
#
#!/bin/sh
set -x
USER_COUNT='wc -l < /etc/passwd'
echo Number of users on node 'hostname' : $USER_COUNT
if test $USER_COUNT -gt §USER-LIMIT
then
    echo USER_COUNT_WARN
    exit 1
else
    echo USER_COUNT_OK
fi
```

Die zu verwendende Symboltabelle soll wie folgt aussehen:

Symbol-Name	Aktueller Wert
USER-LIMIT	100

Daraus ergibt sich die folgende ausführbare Shell-Prozedur:

```
#
# Bourne shell script for checking the number of users
# entered in /etc/passwd.
# If more than 100 entries appear,
# the script will be ended with exit 1.
#
#!/bin/sh
set -x
USER_COUNT='wc -l < /etc/passwd'
echo Number of users on node 'hostname' : $USER_COUNT
if test $USER_COUNT -gt 100
then
    echo USER_COUNT_WARN
    exit 1
else
    echo USER_COUNT_OK
fi
```

Anmerkung: Alle zur Aktivierungszeit durch die Natural MACRO-Funktionalität aufgebauten Jobkontrollanweisungen können durch den Benutzer noch solange modifiziert werden, bis der Job tatsächlich gestartet wird. Natürlich sind diese Modifikationen dann nur für den aktuellen Netzwerklauf gültig.

Editor für System-Objekte

Mit der Editor-Funktionalität in Entire Operations kann der Benutzer folgende Objekte erstellen, anzeigen oder bearbeiten:

- die Jobkontrollanweisungen (JCL) von Jobs, entweder aus der **Master-Datenbank** (und somit beliebigen externen Speicherquellen) oder aus der **aktiven Datenbank**.

  Änderungen an den Jobkontrollanweisungen von aktiven Jobs beeinflussen nur den aktuellen Netzwerklauf, sie haben keine Auswirkungen auf die Master-Datenbank
- Natural-Programme und Benutzerroutinen;
- Entire Operations MAC (Makro)-Jobs;
- Online Dokumentation von Netzwerken, Jobs oder Ereignissen innerhalb von Jobs (Notizzettel-Informationen);
- Job-Protokolle (nur zum Anzeigen);
- Job-Ausgabelisten (nur zum Anzeigen).

Der Benutzer von Entire Operations ist somit zum Beispiel in der Lage, so unterschiedliche Daten wie UNIX-Prozeduren, CA-LIBRARIAN-Dateien oder LMS-Dateien mit einem einzigen Editor zu bearbeiten.

Der Benutzer kann den Editor aufrufen, indem er die Edit-Funktion im Verwaltungsbildschirm des entsprechenden Objekts auswählt.

Der Editor stellt eine umfassende Funktionalität z/OS/ISPF-ähnlicher Kommandos zur Verfügung. Diese sind dem Typ des Objekts angepasst, welches editiert werden soll: zum Beispiel sichert und katalogisiert das Kommando `STOW` ein Natural-Programm, es sollte aber für Natural-MACRO-Programme nicht verwendet werden. Natural-MACRO-Programme werden mit dem `MACRO`-Kommando gesichert, vorverarbeitet und katalogisiert.

An textverarbeitenden Funktionen bietet der Editor Zentrierung, physische und logische Tabulator-Benutzung und Text-Überlagerung.

Berichtsfunktionen

Entire Operations bietet eine große Auswahl an Berichten („Reports“) an, um die Arbeit mit dem System auf allen Ebenen zu unterstützen. Der Benutzer kann die Berichtsfunktionen aufrufen, indem er die entsprechende Option auswählt oder das Direktkommando `REPORTS` in der Kommandozeile benutzt.

Berichte decken folgende Bereiche ab:

- Information über alle Jobs, selektierbar nach abgebrochenen, beendeten oder nicht gestarteten Jobs. Datumsintervalle sowie Netzwerknamen können zur weiteren Einschränkung der Suche angegeben werden. Die für diese Objekte erstellten Berichte können alle Ereignisse, Aktivierungszeit, Meldungen, Job-Ende-Status usw. enthalten. Alle Job-Auswertungen sind nach Protokollierungszeitpunkten sortiert.
- Netzwerk-Information, entweder in der Form eines kurzen Überblicks oder als umfassender Bericht, der detaillierte Informationen aller Netzwerk-Komponenten enthält. In allen Netzwerk-Berichts sind Informationen über das Netzwerk selbst, alle Jobs, die Eingabebedingungen und Ressourcen sowie bei Job-Ende-Behandlung inklusive der Ausgabebedingungen enthalten. Diese ausführlichere Beschreibung beinhaltet zudem auch alle Prosa-Beschreibungen auf Netzwerk-, Job- und Ereignis-Ebene.
- Zeitplanübersicht für ausgewählte oder alle Netzwerke, inklusive der Liste aller Jobs, die innerhalb eines bestimmten Zeitfensters eingeplant sind. Diese Auswertung kann entweder für bereits vergangene Perioden angefordert werden, um z.B. alle nicht-erfolgten Netzwerk-Aktivierungen aufgelistet zu bekommen, oder aber für zukünftige Produktionsperioden, um Informationen für deren Vorhersage und Planung zu gewinnen.
- Vergleich, ob alle Symbole einer Tabelle in einer anderen Tabelle vorhanden sind. Außerdem werden die Definitionen von gleichnamigen Symbolen verglichen.

- Vergleich, ob alle Jobs eines Netzwerks in einem anderen Netzwerk vorhanden sind. Außerdem werden die Definitionen der Netzwerke und die der gleichnamigen Jobs verglichen.
- Knoten-Übersicht, liefert eine nach Auswahlkriterien einschränkbare Übersicht über die in Entire Operations verwendeten Knoten (Server) sowie ausführliche Angaben zu den aufgelisteten Knoten.
- Eine nach Auswahlkriterien (Eigentümer, Netzwerk, Version) einschränkbare Übersicht über die Verwendung von Unternetzwerken.

Alle Berichte sind online verfügbar, die Daten können aber auch gedruckt werden. Hiermit ist eine einfache Möglichkeit einer Langzeit-Dokumentation gegeben.

Die oben aufgeführten Funktionen können auch im Batch-Betrieb ausgeführt werden.

Mit Hilfe der Entire Operations Import/Export-Funktionen kann der Inhalt der Master-Datenbank in eine Einfachdatei (Flat File) entladen werden. Neben der eigentlichen Verwendung für Datenmigration und Transport können Sie diese Funktionen auch dazu benutzen, Ihr eigenes Berichtssystem aufzubauen.

Weitere Informationen siehe *Berichte* im *Benutzerhandbuch*.

Cross-Referenzen

Es stehen Online-Funktionen zur Verfügung, die Aussagen über die Verwendung von Objekten in Entire Operations liefern. Dieselben Funktionen können auch im Batch-Betrieb ausgeführt werden.

Weitere Informationen siehe *Cross-Referenzen* im *Benutzerhandbuch*.

Benachrichtigung

Entire Operations kann Nachrichten an verschiedene Stellen versenden. Das Versenden kann durch systeminterne oder benutzerdefinierte Ereignisse ausgelöst werden.

Weitere Informationen siehe *Nachrichten* im *Benutzerhandbuch*.