Konzept und Leistungsumfang

Version 6.0.1

Oktober 2026

ADABAS & NATURAL

This document applies to Version 6.0.1 and all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Copyright © 2021-2026 Software GmbH, Darmstadt, Germany and/or its subsidiaries and/or its affiliates and/or their licensors.

The name Software AG and all Software GmbH product names are either trademarks or registered trademarks of Software GmbH and/or its subsidiaries and/or its affiliates and/or their licensors. Other company and product names mentioned herein may be trademarks of their respective owners.

Detailed information on trademarks and patents owned by Software GmbH and/or its subsidiaries is located at https://softwareag.com/licenses.

This software may include portions of third-party products. For third-party copyright notices, license terms, additional rights or restrictions, please refer to "License Texts, Copyright Notices and Disclaimers of Third-Party Products". For certain specific third-party license restrictions, please refer to section E of the Legal Notices available under "License Terms and Conditions for Use of Software GmbH Products / Copyright and Trademark Notices of Software GmbH Products". These documents are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Use of this software is subject to adherence to Software GmbH's licensing conditions and terms. These terms are part of the product documentation, located at https://softwareag.com/licenses and/or in the root installation directory of the licensed product(s).

Document ID: NOW-ONOPCONCEPTS-601-20261023DE

Inhaltsverzeichnis

Konzept und Leistungsumfang v
1 Über diese Dokumentation 1
Dokumentationskonventionen 2
Online-Informationen und Support 2
Datenschutz 3
I Wozu dient Entire Operations? 5
2 Wozu dient Entire Operations? 7
Einleitung 8
Mehr als ein Scheduler 9
Zusammenfassung der Vorteile 13
Benutzerschnittstellen 13
II Entire Operations-Objekte 19
3 Entire Operations-Objekte 21
Beziehungen zwischen Entire Operations-Objekten 22
Benutzer und Eigentümer 24
Job-Netzwerke 26
Jobs 28
Logische Bedingungen 29
Mailboxen, Versenden von Nachrichten 33
Ressourcen 37
Kalender 37
Zeitpläne 38
Symboltabellen und Symbole 39
Laufnummer 40
Objekt-Versionierung 40
Betriebssystem-Server-Knoten 41
III Entire Operations-Komponenten 43
4 Entire Operations-Komponenten 45
Master-Datenbank 46
Aktive Datenbank 47
Externe Jobkontrollanweisungen (JCL) 50
Entire Operations-Monitor 50
Betriebssystem 52
Protokollierungsfunktionalität (Logging) 52
Berichtsfunktionalität (Reporting) 53
Editor 53
IV Entire Operations-Funktionen 55
5 Entire Operations-Funktionen 57
..... 58
Verwaltung der Job-Netzwerke 58
Zeitplanung für Job-Netzwerke 58
Job-Verwaltung 58
Job-Zeitplanung 59

Kalender-Definition 59
Verwaltung der Bedingungen 60
Job-Ende-Prüfung und -Aktionen 61
Dynamische JCL-Generierung (MACRO-Funktionalität) 62
Editor für System-Objekte 67
Berichtsfunktionen 68
Cross-Referenzen 69
Benachrichtigung 70

Konzept und Leistungsumfang

Wozu dient Entire Operations?	Vorstellung der besonderen Merkmale von Entire Operations und des Entire Operations GUI Client.
Entire Operations-Objekte	Beschreibung der Entire Operations-Objekte wie zum Beispiel Eigentümer und Benutzerkennungen, Job-Netzwerke, Jobs, logische Bedingungen, Mailboxen, Ressourcen, Kalender, Zeitpläne, Symboltabellen.
Entire Operations-Komponenten	Informationen zu den Entire Operations-Komponenten wie zum Beispiel Master-Datenbank, externe Jobkontrollanweisungen, aktive Datenbank, Monitor, Betriebssystem, Benutzerschnittstelle, Protokollierungsfunktionalität („Logging“), Berichtsfunktionalität („Reporting“), Editor.
Entire Operations-Funktionen	Menü-unterstützte Entire Operations-Funktionen, die Ihnen die Definition von Objekten innerhalb des Systems und die Kontrolle und Überwachung der Verarbeitung von Job-Netzwerken erlauben.

1 Über diese Dokumentation

- Dokumentationskonventionen 2
- Online-Informationen und Support 2
- Datenschutz 3

Dokumentationskonventionen

Konvention	Beschreibung
Fettschrift	>Kennzeichnet Elemente auf einem Bildschirm.
`Nichtproportionale Schrift`	Kennzeichnet Namen und Orte von Diensten im Format *`Ordner.Unterordner.Dienst`*, Programmierschnittstellen (APIs), Namen von Klassen, Methoden und Properties in Java.
`Kursivschrift`	Kennzeichnet: Variablen, für die Sie situations- oder umgebungsspezifische Werte angeben müssen. Neue Begriffe, wenn sie erstmals im Text auftreten. Verweise auf andere Dokumentationsquellen.
`Nichtproportionale Schrift`	Kennzeichnet: Text, den Sie eingeben müssen. Meldungen, die vom System angezeigt werden. Programmcode.
{ }	Zeigt eine Reihe von Auswahlmöglichkeiten an, von denen Sie eine auswählen müssen. Geben Sie nur die innerhalb der geschweiften Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole { } ein.
|	Trennt zwei sich gegenseitig ausschließende Auswahlmöglichkeiten in einer Syntaxzeile voneinander ab. Geben Sie eine der Auswahlmöglichkeiten ein. Geben Sie nicht das Symbol | ein.
[ ]	Zeigt eine oder mehrere Optionen an. Geben Sie nur die innerhalb der eckigen Klammern vorhandenen Informationen ein. Geben Sie nicht die Klammersymbole [ ] ein.
...	Zeigt an, dass Sie mehrere Auswahlmöglichkeiten desselben Typs eingeben können. Geben Sie nur die Informationen ein. Geben Sie nicht die drei Auslassungspunkte (...) ein.

Online-Informationen und Support

Produktdokumentation

Sie finden die Produktdokumentation auf unserer Dokumentationswebsite unter **https://documentation.softwareag.com**.

Zusätzlich können Sie auch über **https://www.softwareag.cloud** auf die Dokumentation für die Cloud-Produkte zugreifen. Navigieren Sie zum gewünschten Produkt und gehen Sie dann, je nach Produkt, zu „Developer Center“, „User Center“ oder „Documentation“.

Produktschulungen

Sie finden hilfreiches Produktschulungsmaterial auf unserem Lernportal unter **https://knowledge.softwareag.com**.

Tech Community

Auf der Website unserer Tech Community unter **https://techcommunity.softwareag.com** können Sie mit Experten der Software AG zusammenarbeiten. Von hier aus können Sie zum Beispiel:

- Unsere umfangreiche Wissensdatenbank durchsuchen.
- In unseren Diskussionsforen Fragen stellen und Antworten finden.
- Die neuesten Nachrichten und Ankündigungen der Software AG lesen.
- Unsere Communities erkunden.
- Unsere öffentlichen Repositories auf GitHub and Docker unter **https://github.com/softwareag** und **https://hub.docker.com/publishers/softwareag** besuchen und weitere Ressourcen der Software AG entdecken.

Produktsupport

Support für die Produkte der Software AG steht lizenzierten Kunden über unser Empower-Portal unter **https://empower.softwareag.com** zur Verfügung. Für viele Dienstleistungen auf diesem Portal benötigen Sie ein Konto. Wenn Sie noch keines haben, dann können Sie es unter **https://empower.softwareag.com/register** beantragen. Sobald Sie ein Konto haben, können Sie zum Beispiel:

- Produkte, Aktualisierungen und Programmkorrekturen herunterladen.
- Das Knowledge Center nach technischen Informationen und Tipps durchsuchen.
- Frühwarnungen und kritische Alarme abonnieren.
- Supportfälle öffnen und aktualisieren.
- Anfragen für neue Produktmerkmale einreichen.

Datenschutz

Die Produkte der Software AG stellen Funktionen zur Verarbeitung von personenbezogenen Daten gemäß der Datenschutz-Grundverordnung (DSGVO) der Europäischen Union zur Verfügung. Gegebenenfalls sind in der betreffenden Systemverwaltungsdokumentation entsprechende Schritte dokumentiert.

I Wozu dient Entire Operations?

2 Wozu dient Entire Operations?

- Einleitung 8
- Mehr als ein Scheduler 9
- Zusammenfassung der Vorteile 13
- Benutzerschnittstellen 13

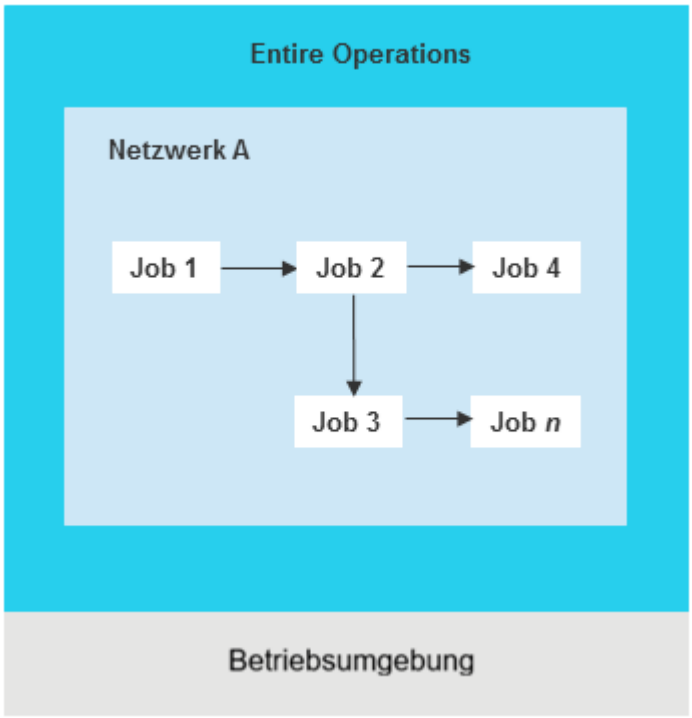
Einleitung

Entire Operations (Produktcode: NOP) ist das Softwaresystem der Software GmbH für die automatisierte Steuerung, Überwachung und Planung von **Job-Netzwerken**. Es bietet alle erforderlichen Funktionen zum Definieren beliebige Arten von Hintergrundverarbeitungen.

Entire Operations bietet komfortable zeichenbasierte (CUI) und grafische **Benutzungsoberflächen** und besitzt alle Merkmale einer modernen, aufgabenorientierten Benutzerführung.

Für den Einsatz von Entire Operations sind keine Änderungen am darunterliegenden 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 Security-Systeme 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 Batchjobs und Scripts benutzt Entire Operations definierte Schnittstellen zu den vorhandenen Spooling-Systemen bzw. funktional äquivalenten Teilen des Betriebssystems selbst.

Mehr als ein Scheduler

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

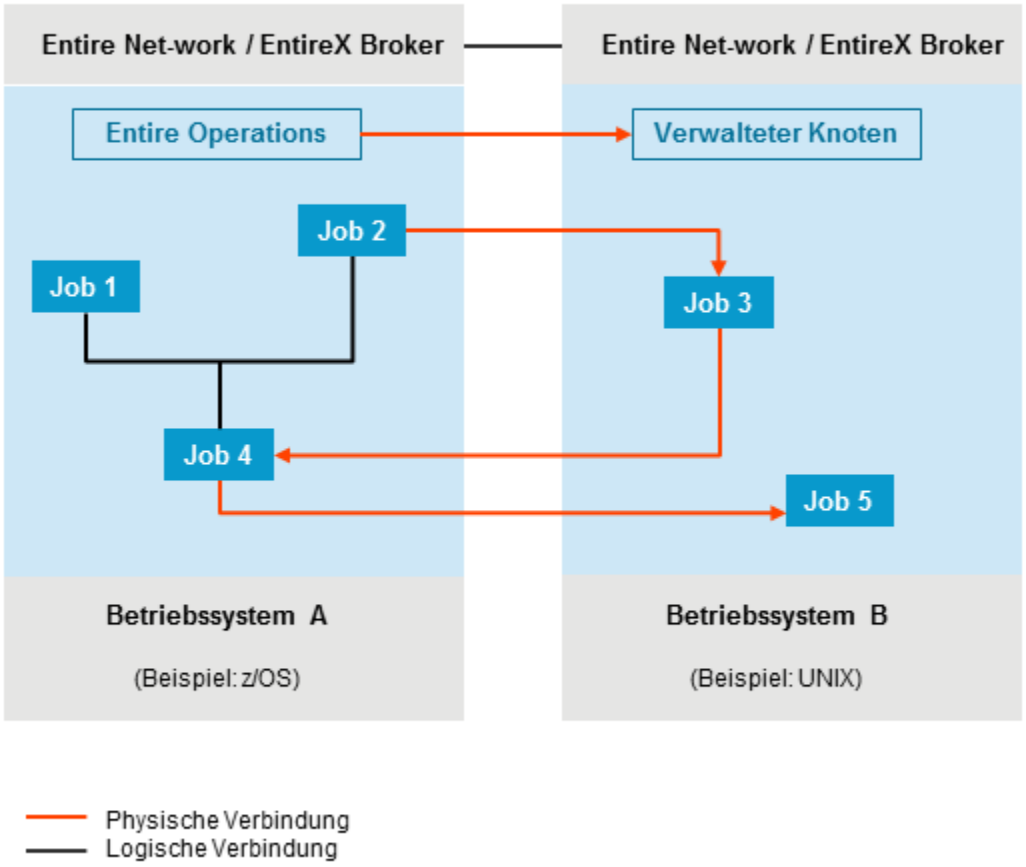
- Steuerung über Betriebssystemgrenzen hinweg
- Entire Operations in einer Konfiguration mit mehreren Betriebssystemen
- Flexible Prozesssteuerung
- Integration anderer Anwendungen
- Einbeziehung von Fachpersonal in die automatisierte Ablaufsteuerung

Steuerung über Betriebssystemgrenzen hinweg

- **Großrechnerumgebungen:**
  Entire Operations kann in allen TP-Umgebungen der Betriebssysteme z/OS und BS2000 eingesetzt werden, es unterstützt insbesondere Com-plete, CICS, TSO, IMS, *open*UTM und TIAM.
- **UNIX:**
  Entire Operations steuert Produktionsprozesse auf UNIX-Systemen (zum Beispiel HP-UX, SINIX RM, AIX, Sun Solaris, Linux) in Verbindung mit einer Großrechnerumgebung. Sie können aber auch Ihre Produktionsumgebung auf dem Mainframe 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 Betriebssystemumgebungen 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 Konfiguration mit mehreren Betriebssystemen



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.

Flexible 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 und vor allem 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.

Integration anderer Anwendungen

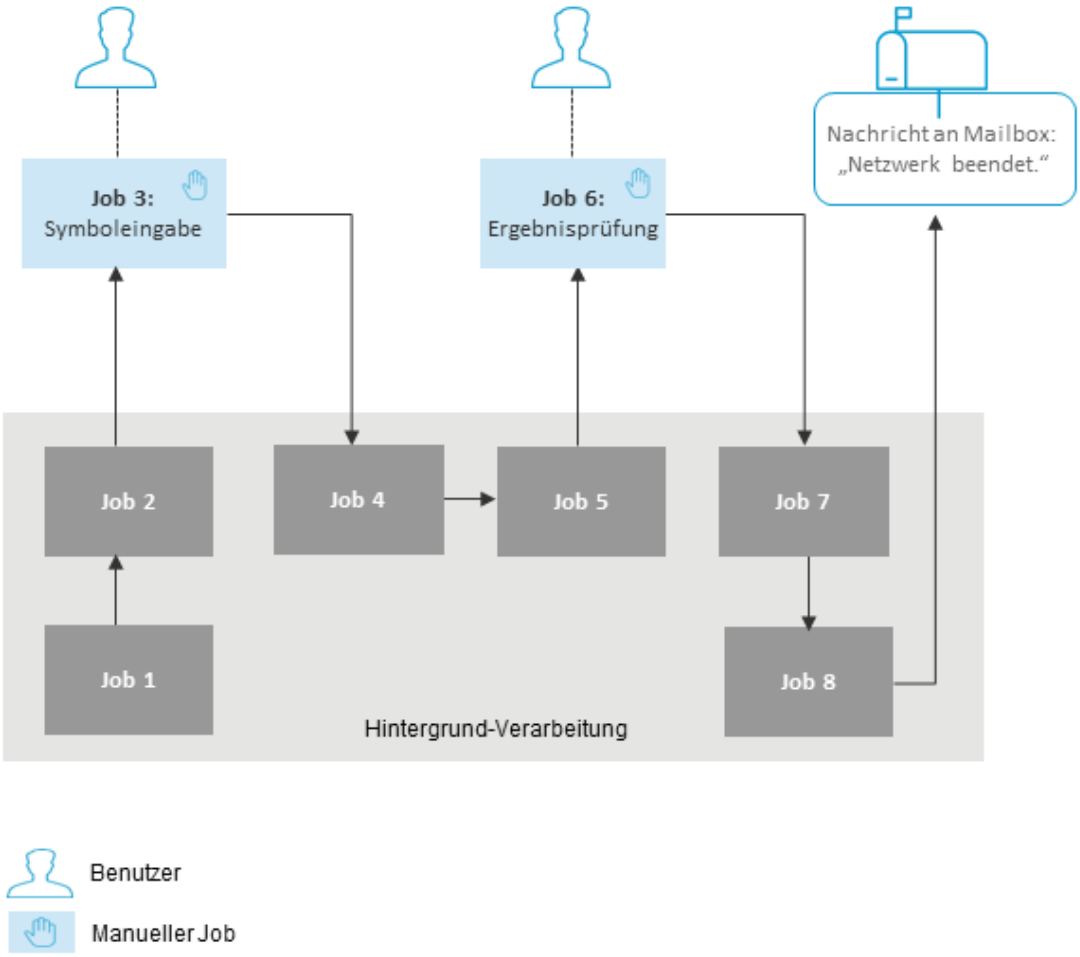
Wer hat sich nicht schon darüber geärgert, dass man zur Erledigung seiner Arbeit mehrere Anwendungen oder Programmsysteme braucht. Hier waren dann oftmals langwierige (und langweilige) An- und Abmeldevorgänge vonnöten oder ein übergeordneter Session-Manager notwendig (was den Ressourcenverbrauch nicht 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 programmtechnischen Integrationsfähigkeit bietet Entire Operations die Möglichkeit, Fachpersonal über ein integriertes Mailbox-Konzept am Produktionsprozess interaktiv zu beteiligen.



Zu selbstdefinierten Zeitpunkten während der Verarbeitung von Hintergrundprozessen können Nachrichten oder aber Eingabeaufforderungen (Symbol) 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 nach Aufforderung ein Symbol 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 der Vorteile

Mit dem Einsatz von Entire Operations als automatisiertes Ablaufsteuerungssystem sichern Sie sich die folgenden grundlegenden Vorteile:

- Transparente Unterstützung verschiedener Computer-Knoten, und dies sogar in heterogenen Umgebungen mit z/OS-, BS2000-, UNIX- und Windows-Betriebssystemen.
- Verfügbarkeit in vielen Großrechner- und UNIX-Umgebungen, die von Natural unterstützt werden, zum Beispiel Com-plete, CICS, TSO, *open*UTM und TIAM.
- Deutsche und englische Benutzungsoberfläche verfügbar.
- Existierende Jobkontrollanweisungen (JCL) laufen unverändert unter der Kontrolle von Entire Operations.
- Keine Modifikationen am Betriebssystem nötig.
- Einsatz dynamisch aufgebauter Jobkontrollanweisungen bzw. Scripts, dadurch Zugriff auf neueste Informationen des Betriebssystems oder auf jedes verfügbare Datenbanksystem zur Ausführungszeit der Job-Netzwerke.
- Einbeziehung von Online-Benutzern in den Batch-Produktionsprozess durch das Konzept der **Mailboxen**.
- Offene Schnittstelle für Benutzer-Anwendungen: Informationen aus Entire Operations können in jede Geschäftsanwendung integriert werden, Benutzer können den erforderlichen Input für tägliche oder zukünftige Produktionsläufe eingeben.

Benutzerschnittstellen

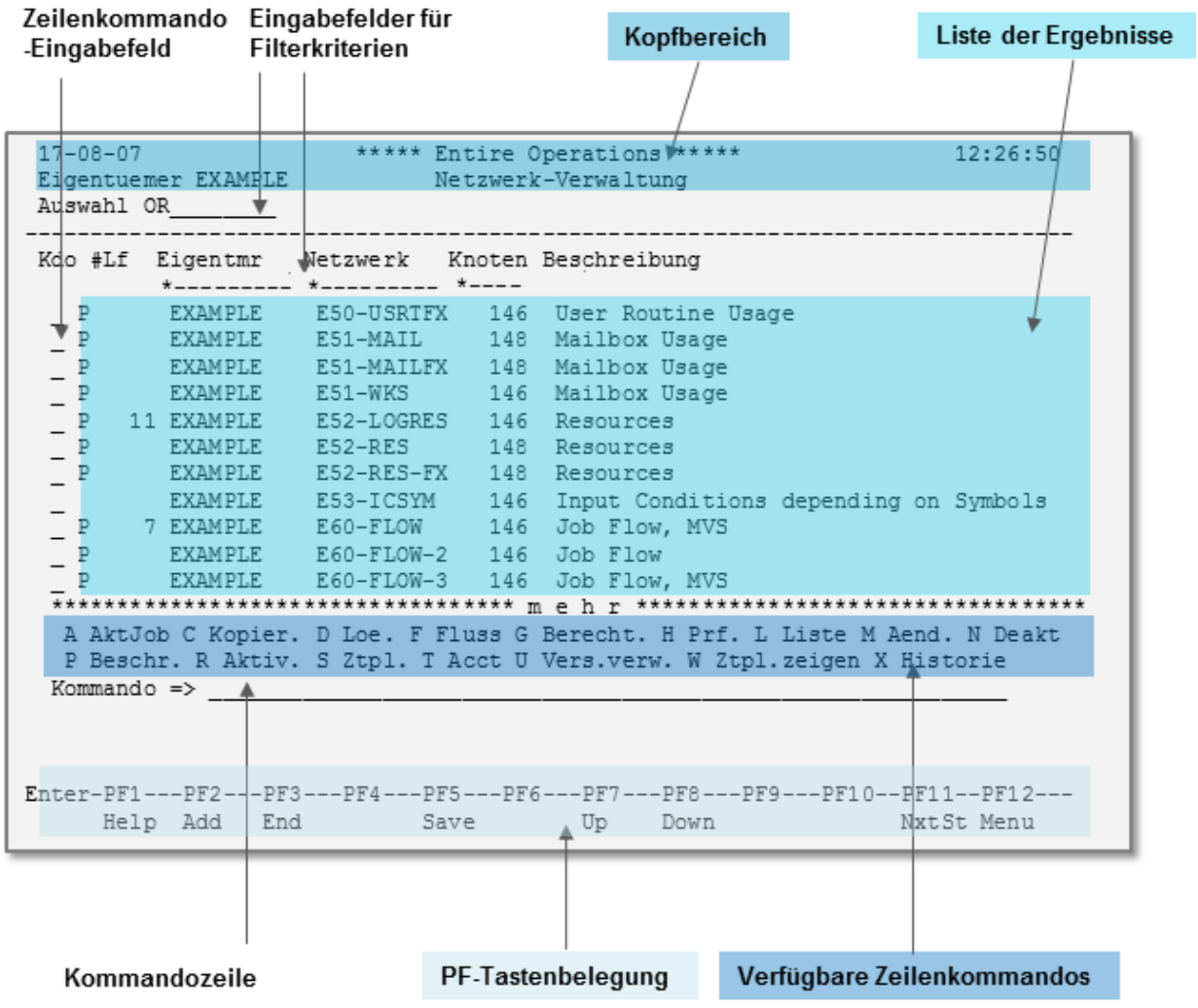
Entsprechend der Plattform, auf der Entire Operations installiert ist, kommt eine menügesteuerte Zeichen-Benutzeroberfläche oder eine grafische Benutzeroberfläche zur Anwendung.

- Entire Operations (Produktcode: NOP) für Mainframe- oder UNIX-Plattformen
- Entire Operations GUI Client (Produktcode: OGC5) auf Windows-Plattformen
- Entire Operations Web Client (Produktcode: OGC6) für Web Browser
- Benutzersprache

- Online-Hilfe

Entire Operations (Produktcode: NOP) für Mainframe- oder UNIX-Plattformen

Die wichtigsten Elemente, die für einen zeichenbasierten Bildschirm typisch sind, sind in der folgenden Beispielabbildung eines Netzwerk-Verwaltung-Bildschirms dargestellt:

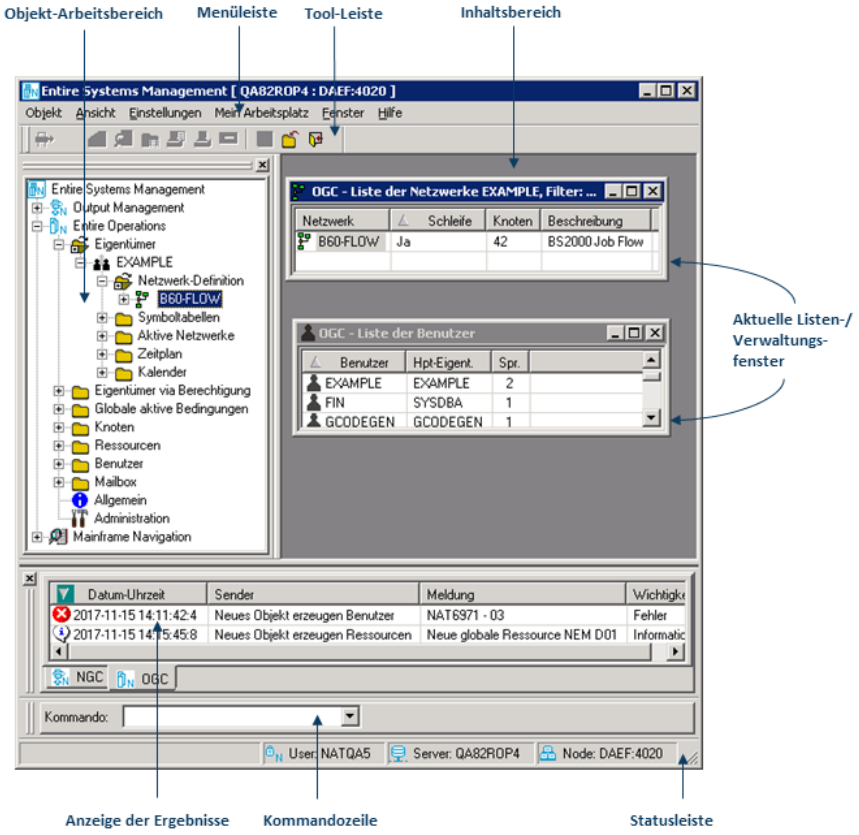


Ausführliche Informationen siehe *Entire Operations-Bildschirme* im *Benutzerhandbuch*.

Entire Operations GUI Client (Produktcode: OGC5) auf Windows-Plattformen

Mit dem Entire Operations GUI Client (Produktcode: OGC) können PC-Benutzer aus einer Windows-Umgebung die Entire Operations-Funktionen auf einem Mainframe oder einer UNIX-Plattform ausführen.

Der Entire Operations GUI Client OGC5 besitzt eine grafische Benutzerschnittstelle mit einem Entire Systems Management-Hauptbildschirm.



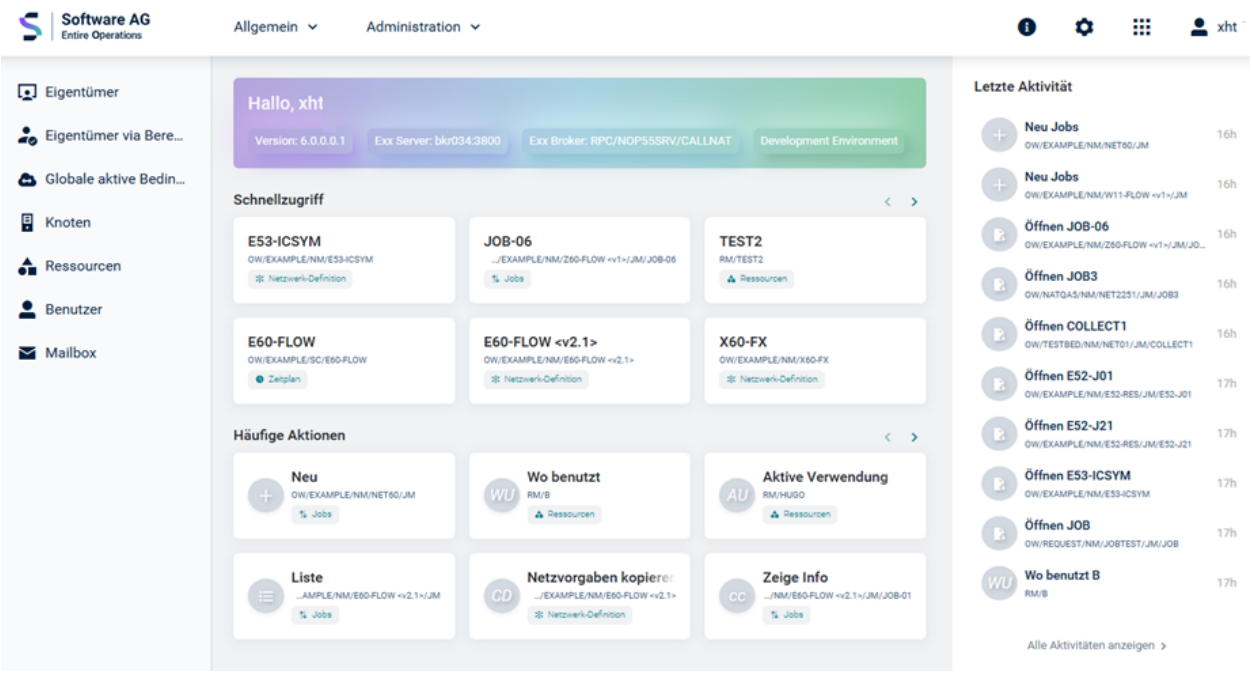
Ausführliche Informationen siehe *Elemente des Entire Systems Management-Hauptbildschirms* im *Benutzerhandbuch*.

Mit dem Entire Operations GUI Client können Job-Netzwerke auch in Form einer Netzplan-Grafik angezeigt und via Kontextmenüs aus der Grafik-Anzeige heraus mit Dialogfunktionen verwaltet werden.

Entire Operations Web Client (Produktcode: OGC6) für Web Browser

Mit dem Entire Operations Web Client können Benutzer aus einer Web-Umgebung die Entire Operations-Funktionen auf den unterstützten Plattformen ausführen. Der Entire Operations Web Client bietet eine grafische Benutzeroberfläche.

Beim Entire Operations Web Client erscheint nach dem Aufrufen zunächst die Startseite (Beispiel):



Ausführliche Informationen siehe *Bestandteile der Benutzeroberfläche* im *Benutzerhandbuch*.

Mit dem Entire Operations Web Client können Job-Netzwerke auch in Form einer Netzplan-Grafik angezeigt und aus der Netzplan-Anzeige heraus mit Dialogfunktionen verwaltet werden. Weitere Informationen siehe *Beispiele für Netzpläne* im Kapitel *Netzplan verwalten* im *Benutzerhandbuch*.

Benutzersprache

Entire Operations ist auf Englisch und Deutsch verfügbar, dies gilt für alle Elemente der Benutzeroberflächen, Hilfeinformationen und Fehlermeldungen.

Jeder Benutzer kann die Sprache individuell einstellen.

Online-Hilfe

Entire Operations bietet eine umfassende Online-Hilfe, mit der Sie Erläuterungen zu einem einzelnen Feld, einem gesamten Fenster oder einem Element in der Baumansicht des Objekt-Arbeitsbereichs anzeigen können.

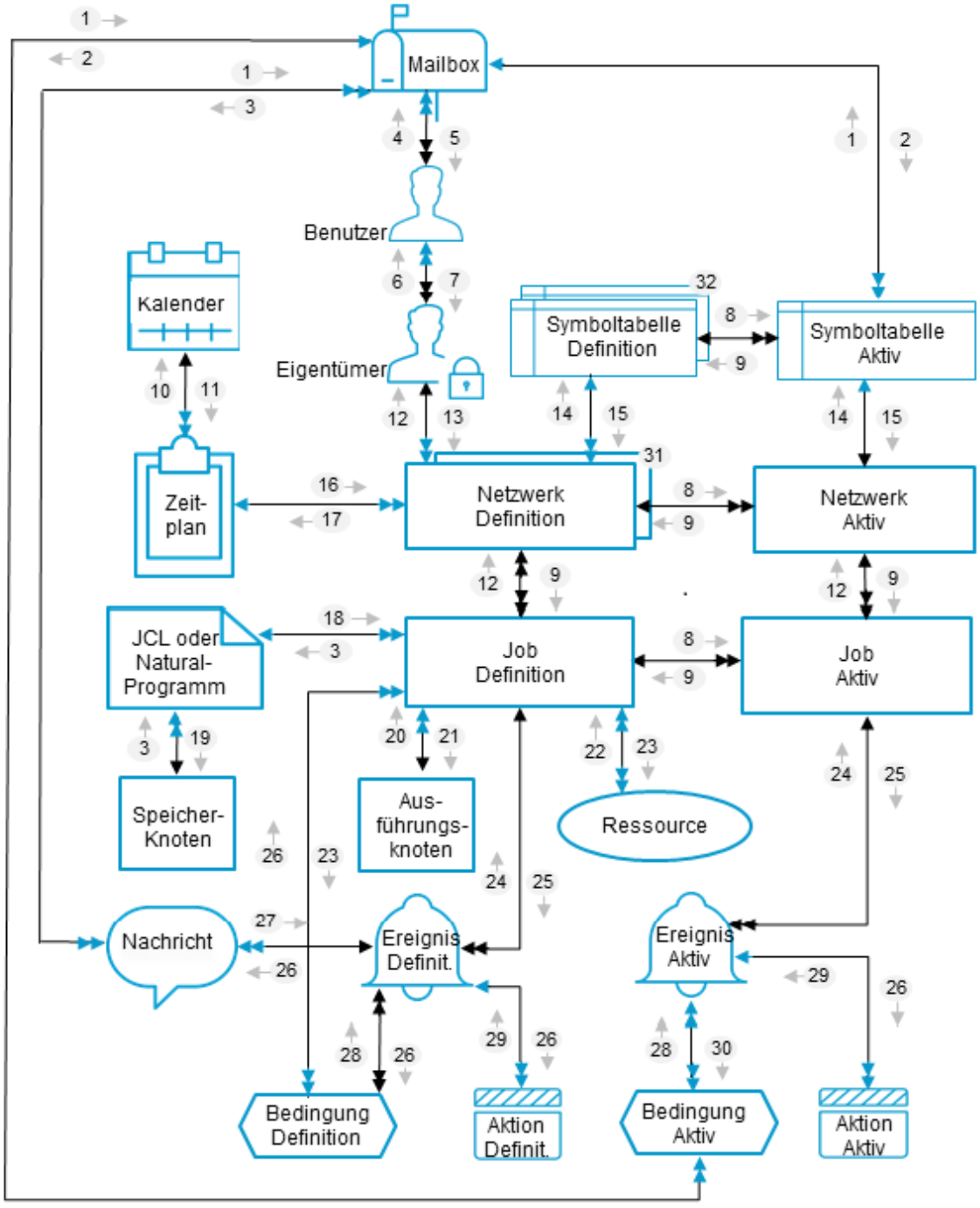
Ausführliche Informationen siehe *Online-Hilfe und Systemmeldungen* im *Benutzerhandbuch*.

II Entire Operations-Objekte

3 Entire Operations-Objekte

- Beziehungen zwischen Entire Operations-Objekten 22
- Benutzer und Eigentümer 24
- Job-Netzwerke 26
- Jobs 28
- Logische Bedingungen 29
- Mailboxen, Versenden von Nachrichten 33
- Ressourcen 37
- Kalender 37
- Zeitpläne 38
- Symboltabellen und Symbole 39
- Laufnummer 40
- Objekt-Versionierung 40
- Betriebssystem-Server-Knoten 41

Beziehungen zwischen Entire Operations-Objekten



Legende:

▶	darf nur eins sein
▶	keines oder höchstens eins
▶▶	muss mindestens eins sein, kann mehr als eins sein
▶▶	keines oder eins oder viele
→	Richtungsanzeige
1	wird gesendet an
2	enthält Aufforderung für
3	enthält
4	verwendet oder kann verwenden
5	enthält Nachrichten 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 oder kann abhängig sein von
24	bestimmt Ergebnis von
25	wird geprüft auf
26	triggert
27	wird gesetzt gemäß
28	wird gesetzt oder zurückgesetzt gemäß
29	wird ausgeführt gemäß
30	setzt
31	Ein Objekt „Netzwerk-Definition“ kann mehrere Versionen haben (siehe **Objekt-Versionierung**).
32	Ein Objekt „Netzwerk-Definition“ kann mehrere Versionen haben (siehe **Objekt-Versionierung**).

Benutzer und Eigentümer

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.

Folgende Themen werden behandelt:

- Eigentümer
- Beispiel: Benutzer/Eigentümer-Zuordnung

Verwandte Themen

- *Verwaltung der Benutzer* in der *Systemverwaltung*-Dokumentation
-

Eigentümer

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.

Entire Operations bietet mit dem Konzept des Eigentümers erhöhte Benutzerfreundlichkeit und Zugriffskontrolle. Durch Zuordnung von Eigentümern ermöglicht dieses Konzept die Aufteilung der Job-Netzwerke in Gruppen. In der Benutzerverwaltung ordnet der Systemadministrator einer Benutzerkennung einen Eigentümernamen zu. Dieser Eigentümername wird an jedes Netzwerk automatisch übergeben, das von diesem Benutzer definiert wird. Siehe *Zuordnung Benutzer/Eigentümer verwalten* im Kapitel *Verwaltung der Benutzer* in der *Systemverwaltung*-Dokumentation.

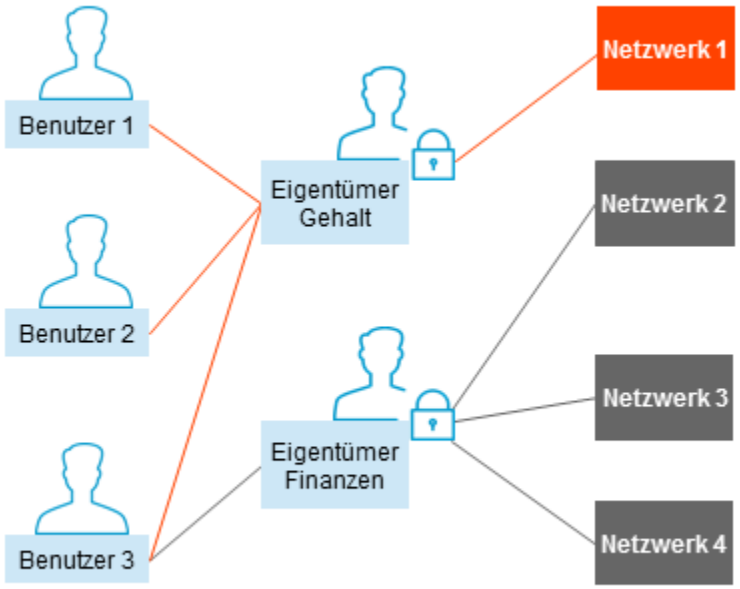
Ein Eigentümer kann dadurch eine Abteilung, ein Projekt oder eine Gruppe zusammengehöriger Job-Netzwerke darstellen. Benutzer, die zu einem bestimmten Eigentümer gehören, können Funktionen nur an Job-Netzwerken ausführen, die mit diesem Eigentümer verknüpft sind.



Anmerkung: In besonderen Fällen kann ein Eigentümer berechtigt sein, auf Netzwerke zuzugreifen, die anderen Eigentümern gehören. Der Eigentümer SYSDBA ist berechtigt, auf die Netzwerke aller Eigentümer zuzugreifen.

Beispiel: Benutzer/Eigentümer-Zuordnung

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



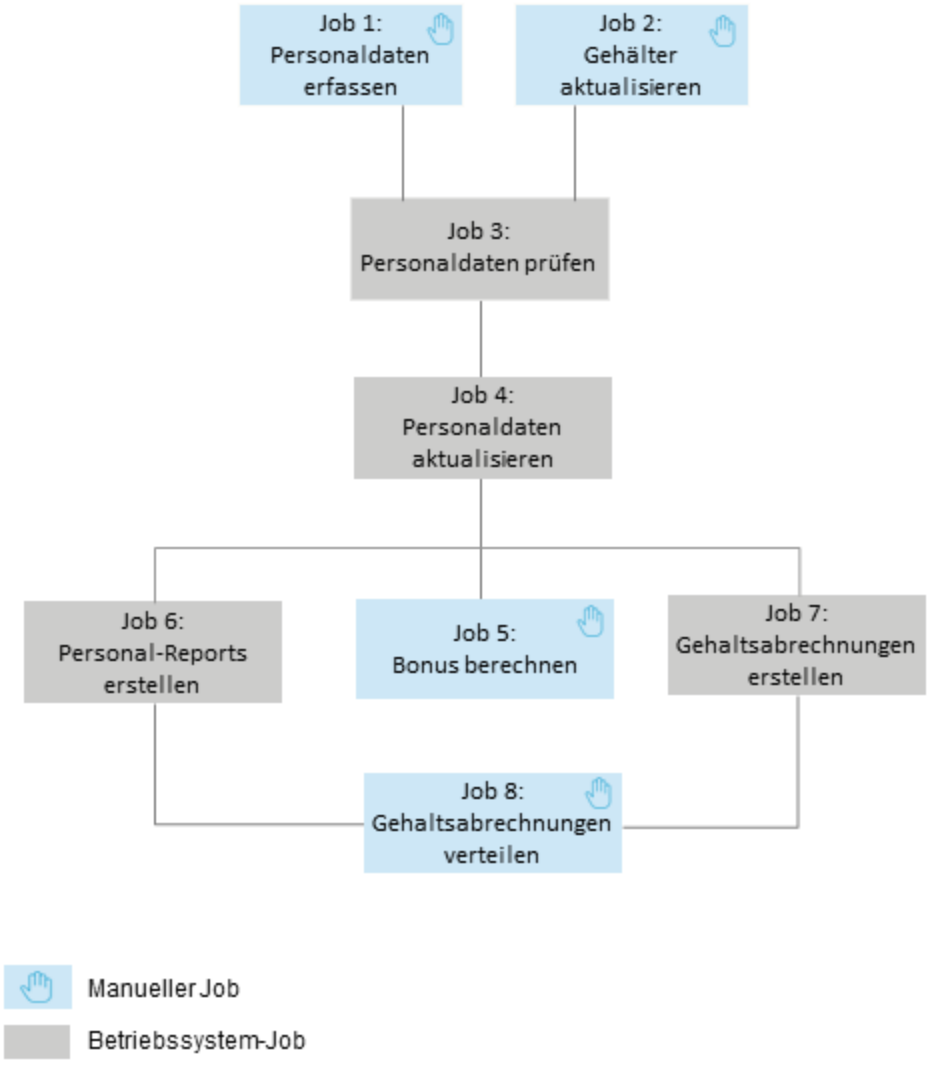
Jeder der drei dargestellten Benutzer kann alle Netzwerke verwalten, die dem Eigentümer *Gehalt* gehören, darüber hinaus ist Benutzer 3 für alle Netzwerke autorisiert, die dem Eigentümer *Finanzen* zugeordnet sind.

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

Job-Netzwerke

Innerhalb Entire Operations umfasst ein Job-Netzwerk eine Gruppe von Jobs, Tasks, Scripts 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 *Netzwerk-Verwaltung* im *Benutzerhandbuch*.

Jobs

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.

Dieser Abschnitt behandelt folgende Themen:

- Jobtypen
- Job-Attribute
- Jobs in einer Mehrrechnerkonfiguration

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

Jobtypen

Entire Operations unterstützt folgende Jobtypen:

- Standard-Jobs des Betriebssystems (z/OS, 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
- 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* im *Benutzerhandbuch*.

Logische Bedingungen

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

Dieser Abschnitt behandelt folgende Themen:

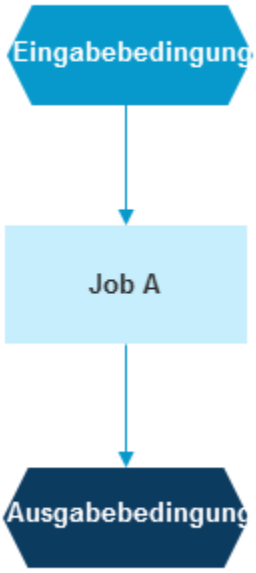
- Eingabe- und Ausgabebedingungen
- Jobs verknüpft durch Eingabe- und Ausgabebedingungen
- Job-Netzwerk mit logischen Bedingungen

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

Informationen zur Prüfung von Bedingungen, siehe Abschnitt *Prüfung von Bedingungen für einen aktiven Job* im *Benutzerhandbuch*.

Eingabe- und Ausgabebedingungen

Die folgende Abbildung zeigt die Beziehung zwischen einer Eingabebedingung, einer Ausgabebedingung und einem Job:

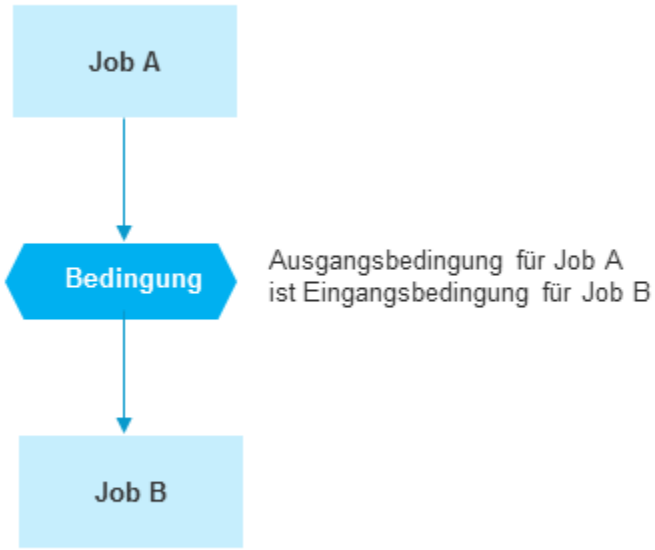


Alle Eingabebedingungen müssen erfüllt sein, bevor ein Job tatsächlich gestartet werden kann (Voraussetzung). Sie können eine beliebige Anzahl von Eingabebedingungen für einen Job definieren.

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 Jobstep-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:

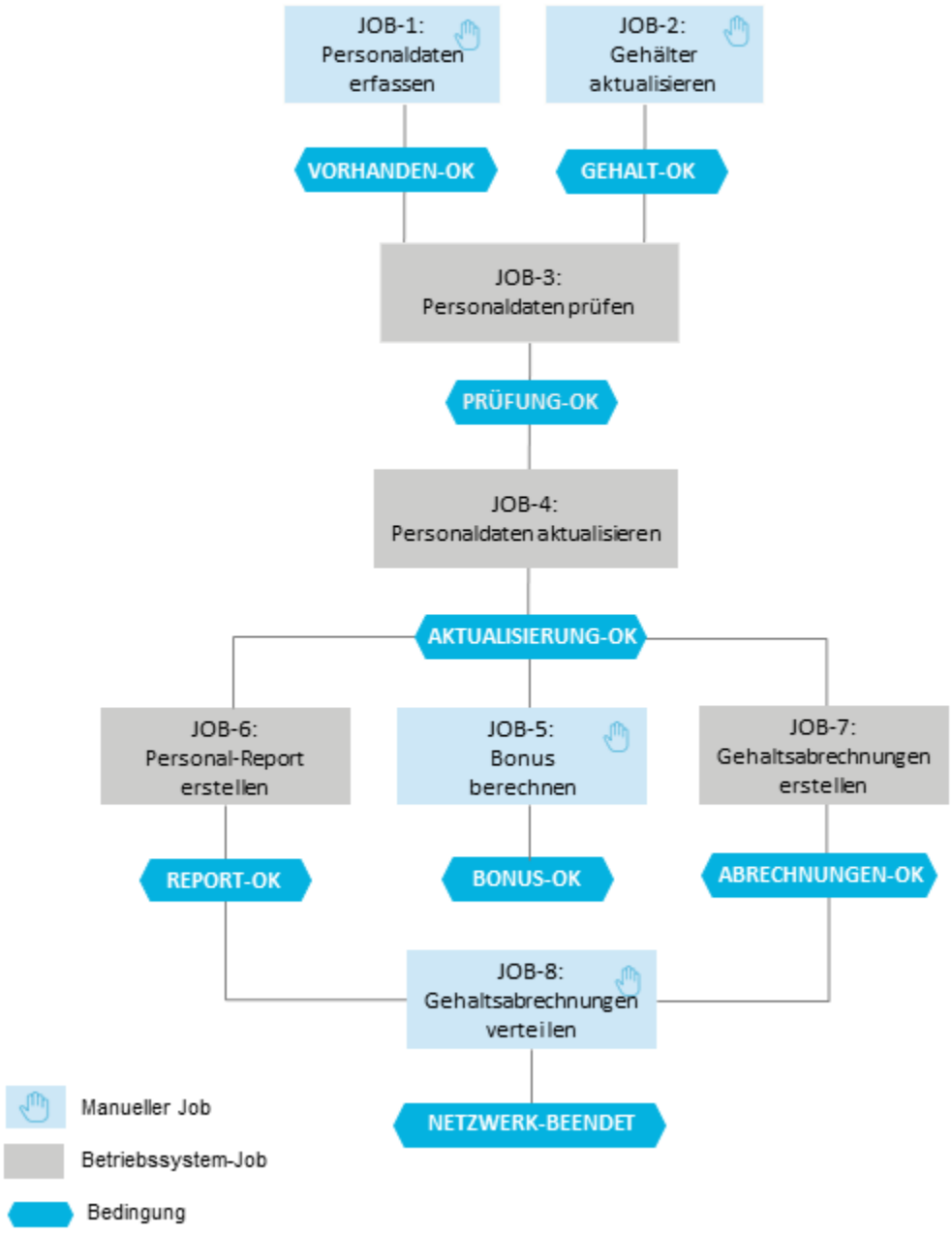


Tritt nun ein bestimmtes Ereignis als Ergebnis von Job 1 ein, so setzt dies die entsprechende Bedingung und signalisiert Entire Operations, dass Job 2 gestartet werden kann.

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:



Jobverknüpfung durch Eingabe- und Ausgabebedingungen

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

Jobnummer	Eingabebedingung	Ausgabebedingung
JOB-1		VORHANDEN-OK
JOB-2		GEHALT-OK
JOB-3	PRÜFUNG-OK	PRÜFUNG-OK
	GEHALT-OK	
JOB-4	PRÜFUNG-OK	AKTUALISIERUNG-OK
JOB-5	AKTUALISIERUNG-OK	BONUS-OK
JOB-6	AKTUALISIERUNG-OK	REPORT-OK
JOB-7	AKTUALISIERUNG-OK	ABRECHNUNGEN-OK
JOB-8	REPORT-OK	NETZWERK-BEENDET
	ABRECHNUNGEN-OK	

Zum Beispiel wird Entire Operations den Job-7 (Gehaltsabrechnung erstellen) solange nicht starten, bis die Eingabebedingung AKTUALISIERUNG-OK erfüllt ist. Diese Bedingung ist auch als Ausgabebedingung für Job-4 definiert.

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

Mailboxen, Versenden von Nachrichten

Innerhalb von Entire Operations dienen Mailboxen dazu, mit Job-Netzwerken und Jobs in Verbindung stehende Nachrichten und Anforderungen an Benutzer und/oder Benutzergruppen zu versenden. Diese Nachrichten können benutzt werden, um Benutzer über den aktuellen Status eines Job-Netzwerks zu informieren oder die Eingabe von Daten anzufordern, die für die weitere Ausführung benötigt werden.

Die zu einer Mailbox aufgelisteten Nachrichten und Anforderungen werden vom **Entire Operations-Monitor** als Resultat automatisch erzeugter Verarbeitungsinformationen oder von Benutzern verfassten Nachrichten und für Jobs definierten Anforderungen erzeugt. Anforderungen können eine Interaktion seitens des Benutzers erfordern, die für das Fortsetzen der Jobverarbeitung nötig ist, z.B. das manuelle Setzen einer Bedingung oder die Reaktion auf eine Symbolabfrage.

Mit Hilfe des Konzepts der Mailboxen kann Entire Operations nicht-CPU-basierte Aufgaben auf die gleiche Weise behandeln wie CPU-basierte:

- Aufgaben 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 informiert werden soll.

Die folgenden Abschnitte erläutern das Konzept der Mailboxen anhand von Beispielen:

- Beispielszenarium - Grundkonzept der Mailboxen
- Beispielszenarium - Konzept für einzelne und mehrere Mailbox-Benutzer

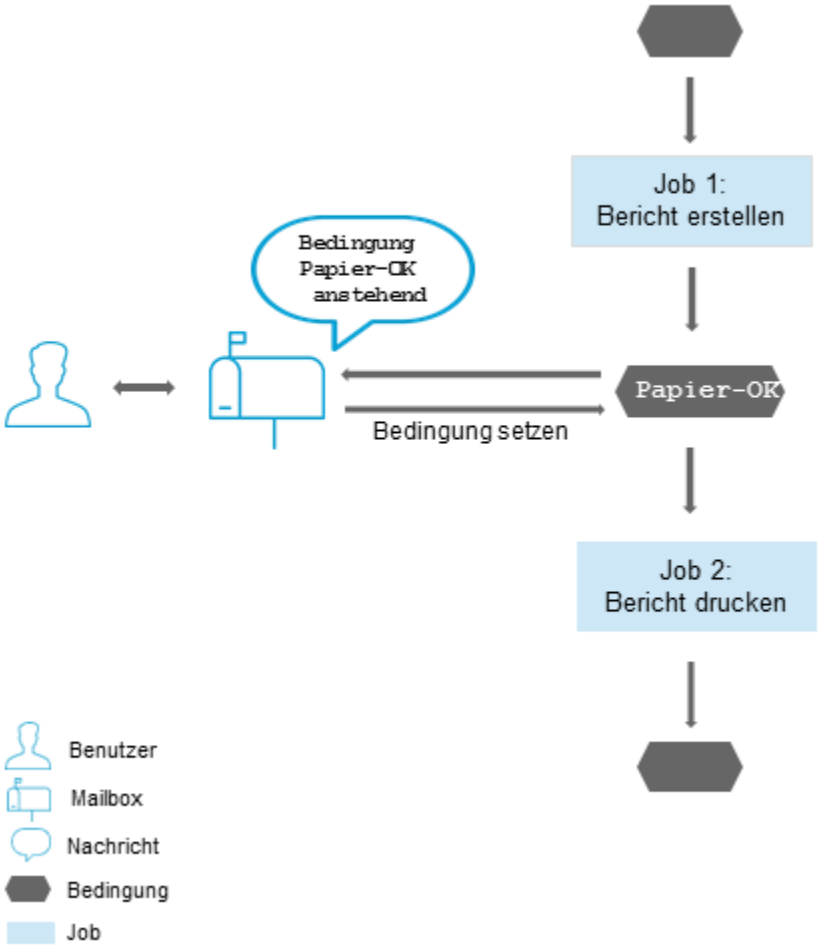
Verwandte Themen:

- *Mailboxen* im *Benutzerhandbuch*
- *Mailbox-Definition* in der *Systemverwaltung*-Dokumentation

Beispielszenarium - Grundkonzept der Mailboxen

Das Konzept der Mailboxen ermöglicht es Ihnen, manuelle Aktionen in das Job-Netzwerk zu integrieren.

Die nachstehende Abbildung zeigt als Beispiel eine Mailbox für „Papiermenge“:



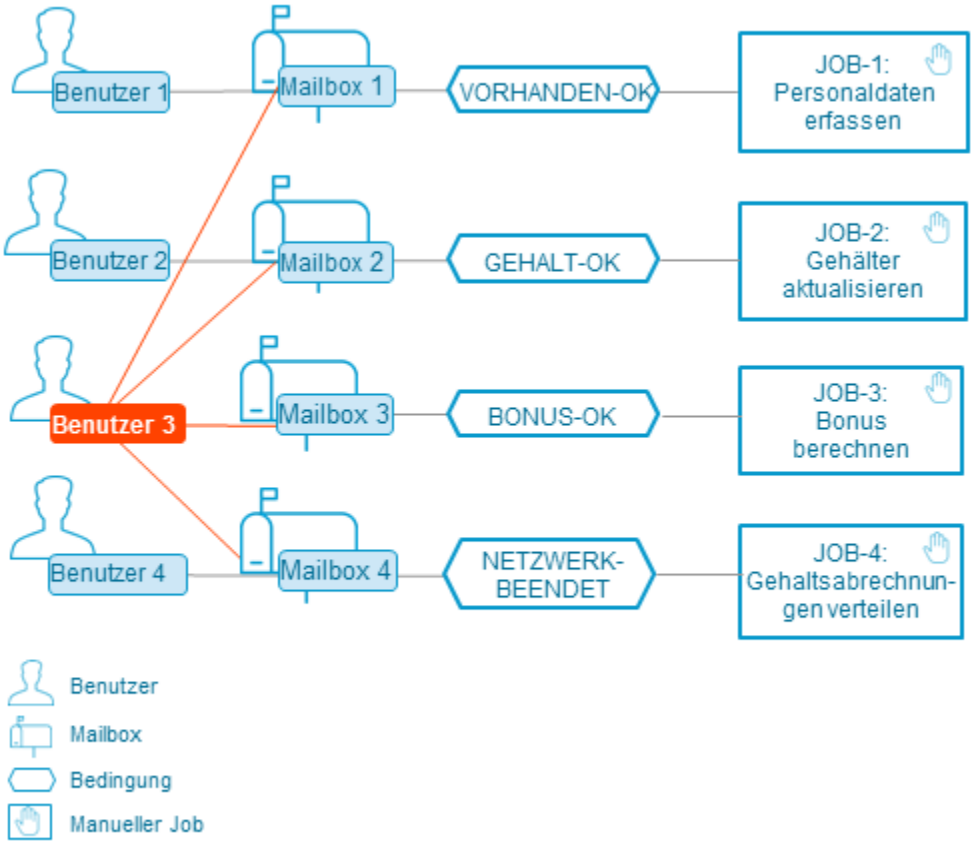
Erläuterungen:

Job 1 „Bericht erstellen“ generiert einen Bericht. Die Bedingung `Papier OK` ist als Eingabebedingung für Job 2 „Bericht drucken“ definiert.

Wenn Sie die Nachricht `Bedingung Papier OK anstehend` empfangen, können Sie die erforderliche Menge Papier verfügbar machen und die Bedingung direkt im Mailbox-Fenster an Ihrem Bildschirm manuell setzen. Entire Operations kann dann mit dem nächsten Job („Bericht drucken“) fortfahren.

Beispielszenarium - Konzept für einzelne und mehrere Mailbox-Benutzer

Die folgende Abbildung veranschaulicht das Konzept der Benutzung von Mailboxen durch einzelne und mehrere Benutzer am Beispiel eines Job-Netzwerks für die Lohnbuchhaltung:



Erläuterungen:

Benutzer 1 (z.B. ein für die Pflege der Personaldaten zuständiger Mitarbeiter) wird benachrichtigt, dass die Bedingung 1 („VORHANDEN-OK“) nicht erfüllt ist.

Der Benutzer kann nun eingreifen, indem er die Personalanwesenheitsdaten vervollständigt und anschließend in seiner Mailbox bestätigt.

Benutzer 3 (z.B. ein Assistent oder Stellvertreter des Personalchefs) wird über jede nicht erfüllte Bedingung in Kenntnis gesetzt und kann so die Abarbeitung des gesamten Job-Netzwerkes überwachen.

Benutzer 3 und Benutzer 4 werden benachrichtigt, wenn das Netzwerk vollständig abgearbeitet ist und die Gehaltsabrechnungsbelege verteilt werden können.

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, das Vorhandensein einer beliebigen katalogisierten Datei oder die tatsächliche Anzahl gerade laufender Jobs überprüfen.

Dem gegenüber stehen die Ressourcen, die nur innerhalb von Entire Operations eine Bedeutung besitzen und die den Verarbeitungsfluss regeln. Der Benutzer kann Ressourcen zur Regulierung des Job-Flusses benutzen oder um die parallele Ausführung von Jobs zu verhindern, wenn alle sonstigen Voraussetzungen (z.B. Zeitfenster oder logische Eingabebedingungen) erfüllt sind. 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.

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.

Weitere Informationen siehe:

- *Vorausgesetzte Ressourcen für einen Job verwalten*
- *Ressourcen-Definitionen verwalten*

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.

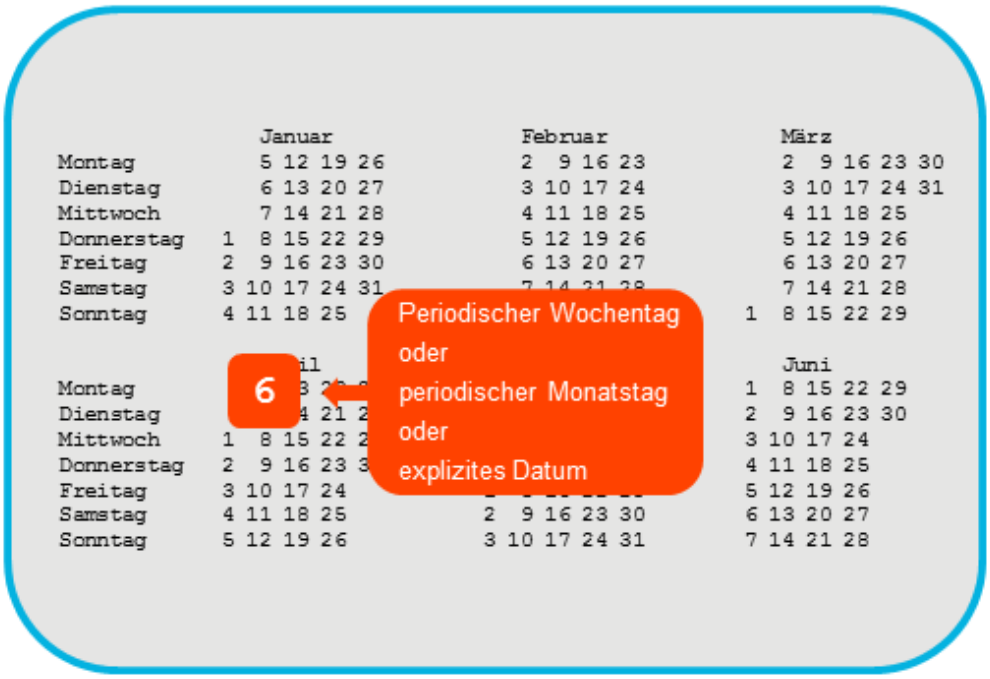
Die Verwendung von Kalendern dient nur einem einzigen Zweck, nämlich zwischen Arbeitstagen und Feiertagen unterscheiden zu können. Entire Operations führt keine Aktivitäten an Feiertagen durch. Sie können festlegen, was mit einem Job-Netzwerk geschehen soll, das an einem solchen Tag eigentlich eingeplant 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 *Allgemeine Informationen zu Kalendern* im Kapitel *Kalender* im *Benutzerhandbuch*.

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 *Allgemeine Aspekte von Zeitplänen* im Kapitel *Zeitpläne* im *Benutzerhandbuch*.

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 von Anwendungsprogrammierungsschnittellen (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 *Verwendung von Symboltabellen und Symbolen* im Kapitel *Symboltabellen und Symbole* im *Benutzerhandbuch*.

Laufnummer

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.

Weitere Informationen siehe *Laufnummer* im *Benutzerhandbuch*.

Objekt-Versionierung

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

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.

Weitere Informationen siehe *Netzwerk-Versionen verwalten* und *Symboltabellen-Versionen verwalten* im Benutzerhandbuch.

Betriebssystem-Server-Knoten

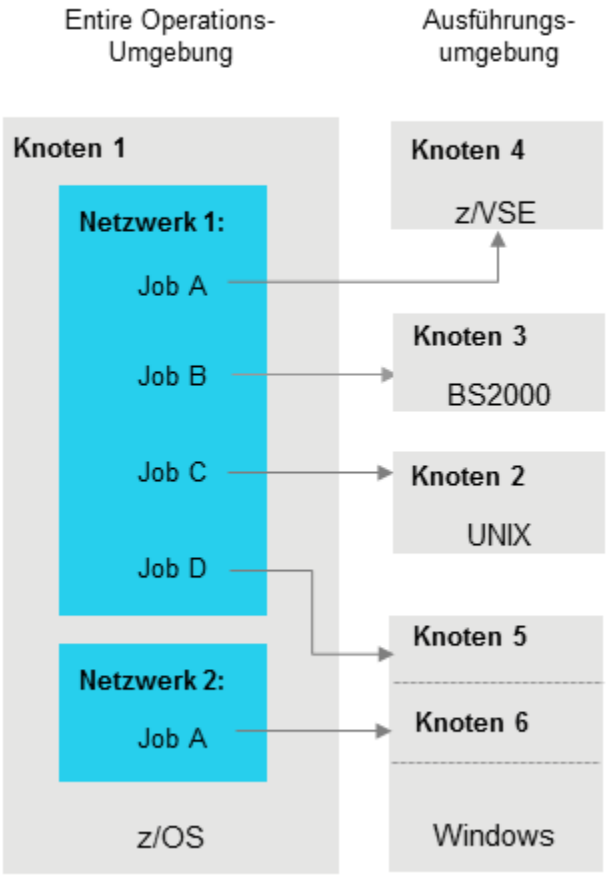
Knoten sind Entire System Server und beziehen sich auf Maschinen, auf denen die Betriebssystem-Anforderungen ausgeführt werden. Knoten unterscheiden sich durch numerische Kennungen, genauso wie Datenbankkennungen zwischen Adabas-Datenbanken unterscheiden. Innerhalb Entire Operations wird jeder Maschine eine Knotennummer zugewiesen. In einer physischen Maschine kann sich mehr als ein Betriebssystem-Server-Knoten befinden.

Auf den Maschinen, die durch Knotenkennungen bezeichnet werden, können verschiedene Ziel-Betriebssysteme laufen. Entire Operations erkennt das Betriebssystem.

Kommunikationswege zwischen ansonsten isolierten Knoten werden von Entire Net-work und EntireX Broker bereitgestellt. Diese Software-AG-Produkte ermöglichen eine transparente Verbindung von Knoten ungeachtet der Art ihrer physischen Verknüpfung.

Bei der Definition eines Job-Netzwerks in Entire Operations können Standard-Knoten für die JCL und für die Ausführung der Jobs angegeben werden. Diese Standard-Knoten können bei jedem Job geändert werden mit der Folge, dass verschiedene Jobs innerhalb desselben Netzwerks auf verschiedenen Maschinen laufen können.

Die folgende Abbildung zeigt ein Beispiel dafür, wie Entire Operations Server und Knoten auf verschiedenen Maschinen und unter verschiedenen Betriebssystemen unterstützen kann:



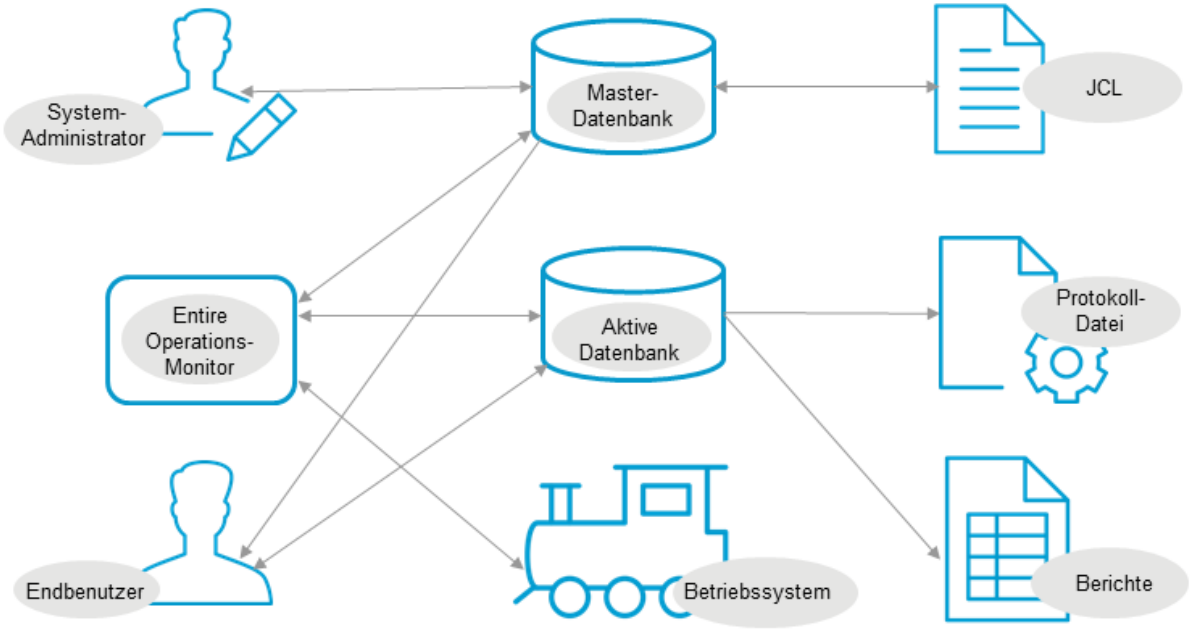
Weitere Informationen siehe *Definition der Knoten* in der *Systemverwaltung*-Dokumentation.

III Entire Operations-Komponenten

4 Entire Operations-Komponenten

- Master-Datenbank 46
- Aktive Datenbank 47
- Externe Jobkontrollanweisungen (JCL) 50
- Entire Operations-Monitor 50
- Betriebssystem 52
- Protokollierungsfunktionalität (Logging) 52
- Berichtsfunktionalität (Reporting) 53
- Editor 53

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 und können online verwaltet werden.

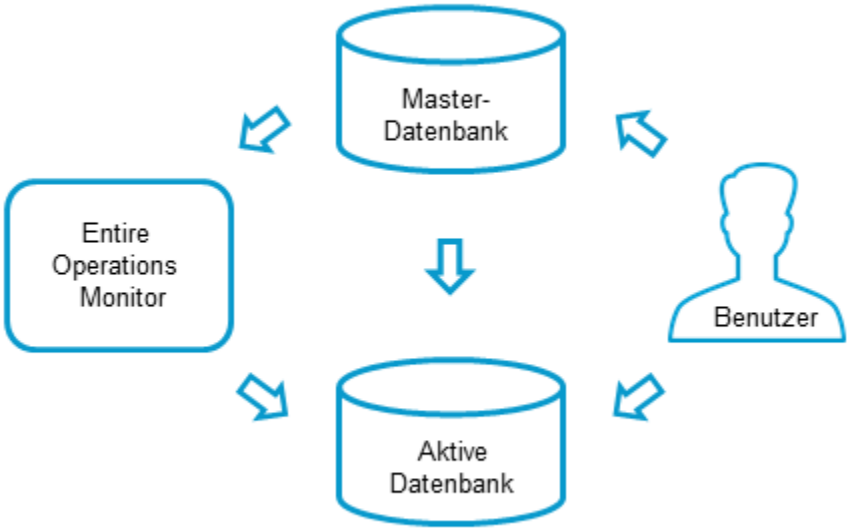
Die Master-Datenbank ist eine Adabas-Datei, was automatisch Benutzer-Synchronisation, Datenintegrität, Datenkomprimierung, automatische Dateierweiterung und automatische Wiederanlauffähigkeit bedeutet.

Alle gespeicherten Objekte können am Bildschirm aus jeder Ablaufumgebung wie zum Beispiel Com-plete, CICS, TSO, TIAM und *open*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

Aktive Datenbank



Immer wenn ein Job-Netzwerk aktiviert wird (siehe *Benutzerhandbuch*), 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 aber durch eine andere **Laufnummer** identifiziert.

Folgende Informationen werden abgelegt:

- Aktuelle Definition eingeplanter Job-Netzwerke zusammen mit ihren aktuellen Symboltabellen.
- Aktive JCL-Bibliothek, was 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.

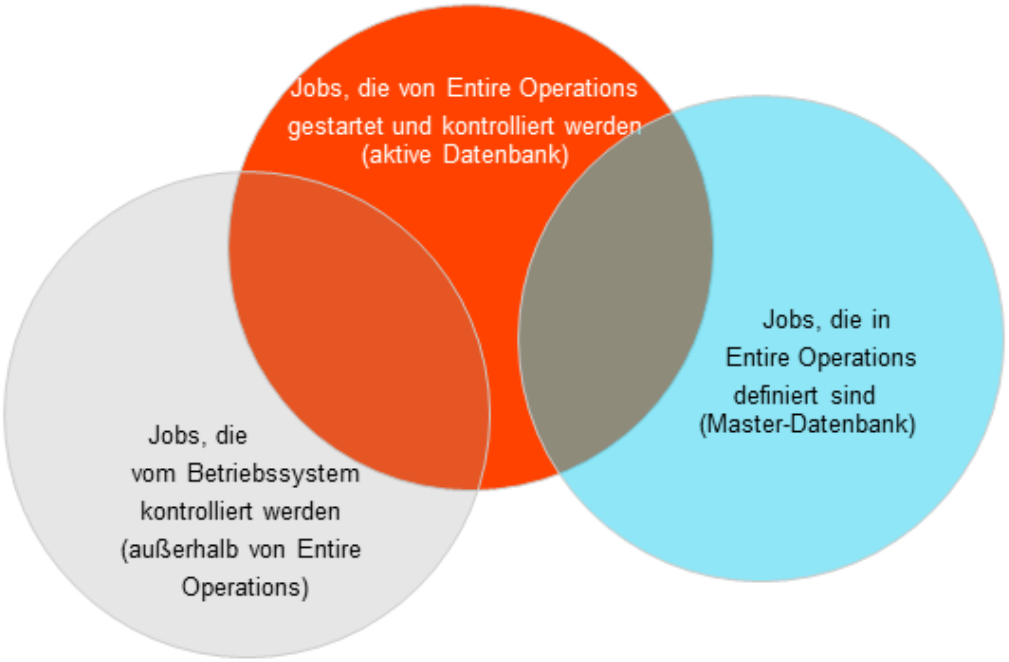
Sie können auf die aktive Datenbank auf die gleiche Art und Weise zugreifen und ihre Informationen modifizieren wie auf die Master-Datenbank. Änderungen an einem Objekt der aktiven Datenbank sind nur für den zugehörigen Netzwerklauf wirksam und haben keinen Einfluss auf die Definitionen der Netzwerke 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

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 des Typs Dummy oder Natural-Programme Entire Operations-Jobs, die nicht dem Betriebssystem übergeben, sondern von Entire Operations ausgeführt werden.

Es wird also zwischen zwei Gruppen von Jobs unterschieden:

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



Der orangene Kreis enthält alle Jobs, die von Entire Operations gestartet und kontrolliert werden. Der blaue Kreis enthält alle Jobs, die für Entire Operations definiert, aber nicht gestartet worden sind. Entire Operations kann keine Jobs steuern, die nicht für Entire Operations definiert sind und die nicht von Entire Operations gestartet werden.

Der Bereich, in dem sich der orangene und der blaue Kreis überschneiden, markiert Entire Operations-Jobs, die bereit sind, von Entire Operations gestartet zu werden.

Jobs außerhalb von Entire Operations (im grauen Kreis) laufen außerhalb der Kontrolle des **Entire Operations-Monitor** auf dem Computer. Andererseits hält Entire Operations Informationen über Jobs bereit, die außerhalb des Einflussbereichs des Betriebssystems laufen.

Der Bereich, in dem sich der graue und der orangene Kreis überschneiden, markiert Entire Operations-Jobs, die an das Betriebssystem übergeben werden.

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.

Die nachfolgenden Abschnitte beschreiben, wie Sie aktive Job-Netzwerke und einzelne aktive Jobs pflegen können.

Externe Jobkontrollanweisungen (JCL)

Mit Entire Operations können Jobkontrollanweisungen unverändert in Job-Netzwerke integriert werden, sie können sogar an ihrem ursprünglichen Speicherort bleiben. Auf z/OS werden sequenzielle Dateien, PDS-, LMS-Bibliotheken sowie die Speicherorte Natural und CA-LIBRARIAN unterstützt. Auf BS2000 kann sich die JCL in SAM- und ISAM-Dateien oder in LMS-Bibliotheken befinden. Auf UNIX-Betriebssystemen können Shell Scripts in die Jobkontrolle von Entire Operations einbezogen werden. Auf Windows können BAT-Dateien benutzt werden.

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

Entire Operations-Monitor

Der Monitor ist das Herz von Entire Operations. Er ist ein komplexes Programm, das periodisch aktiviert wird und die Definitionen in der Master-Datenbank daraufhin überprüft, ob eine Verarbeitung ansteht. Der Monitor aktiviert und verarbeitet Netzwerke und Jobs unter Berücksichtigung ihrer Vorbedingungen und bringt Job-Netzwerke zur Ausführung. Er kontrolliert und steuert laufende Job-Netzwerke auch dann, wenn sich diese auf Knoten verschiedener Rechner befinden.

Der Monitor führt folgende Funktionen durch:

- er aktiviert eingeplante Netzwerke automatisch (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 behält den Überblick über Jobs in den verschiedenen Warteschlangen des Betriebssystems;
- er analysiert den Job-Ende-Status jedes einzelnen Jobs, der beendet wurde, stellt das Eintreten von Ereignissen fest und stößt entsprechende Systemaktionen (Bedingungen setzen, Nachrichten schicken, Programme starten) an;
- er protokolliert alle wichtigen Informationen.

- er bereinigt die **aktive Datenbank**.

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

In technischer Hinsicht können Sie den Monitor auf zwei Arten betreiben, und zwar als eine oder mehrere Subtasks oder als Batch-Task.

Dieses Dokument behandelt folgende Themen:

- Monitor Subtask(s)
- Monitor als Batch-Task betreiben

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

Monitor Subtask(s)

Die einzelnen Funktionen, die der Entire Operations-Monitor durchführt, können auf mehrere Subtasks verteilt werden. Diese Aufgabenverteilung ermöglicht es, Prozesse parallel auszuführen und die Performance zu erhöhen. Die Verteilung der Monitorfunktionen auf Subtasks ist unter z/OS, BS2000 und UNIX möglich. Unter BS2000 und UNIX sind die Monitor-Subtasks separate Prozesse im Betriebssystem.

Monitor Subtask(s) unter z/OS

Sie können den Entire Operations-Monitor als eine oder mehrere Subtasks einer Entire System Server Task in z/OS-Betriebssystemen betreiben.

Die JCL der Entire System Server-Task (XCOM-Knoten) muss für die Anforderungen des Monitors erweitert werden. Außerdem müssen die XCOM-Parameter ergänzt werden. Die REGION-Zuweisung für die Entire System Server-Task muss groß genug sein, um den Monitor aufzunehmen.

Die Vorteile dieser Methode sind:

- Alle Entire System Server-Aufrufe des Monitors an den eigenen Knoten werden lokal bearbeitet, ohne dass eine Inter-PROCESS-Kommunikation stattfindet;
- Der Entire System Server und der Entire Operations-Monitor teilen denselben Adressraum.

Monitor als Batch-Task betreiben

Unter z/OS, BS2000 oder UNIX können Sie den Entire Operations-Monitor als eine eigene Batch-Task betreiben.

Der Monitor kann als normaler Batch-Job laufen. Die Funktionen, die er in diesem Modus zur Verfügung stellt, sind dieselben wie unter einer Entire System Server Subtask. Als Batch-Task erfordert der Monitor jedoch, dass der **Betriebssystem-Server-Knoten** aktiv bleibt, solange er selbst aktiv ist.

Der Systemadministrator kann einen Zeitabstand zwischen Monitorzyklen definieren. Zu Anfang eines Zyklus wird der Monitor aktiviert und prüft dann die Entire Operations-Arbeitswarteschlangen. Hier führt der Monitor alle erforderlichen Aktionen durch: er startet Jobs und analysiert den Job-Ende-Status jedes einzelnen Jobs, der beendet wurde, usw.

Je kürzer die Wartezeit zwischen den Tasks, desto kürzer der Zeitabstand zwischen der Beendigung des Jobs und seiner Job-Ende-Analyse. Diese verkürzte Wartezeit hat allerdings eine zusätzliche Belastung des Systems wegen der häufigen Reaktivierung des Monitors zur Folge.

Betriebssystem

Die Jobverarbeitung wird vom Entire Operations-Monitor auf den zugrundeliegenden Betriebssystemen initiiert: Jobs, Tasks, Scripts oder Natural-Programme werden gestartet. Dabei können mehrere Betriebssysteme gleichzeitig von einem Entire Operations-Monitor bedient werden.

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

Protokollierungsfunktionalität (Logging)

Entire Operations besitzt eine Protokollierungsfunktionalität, mit deren Hilfe jedes Ereignis während einer Netzwerkdurchführung in einem Protokoll („Log“) aufgezeichnet wird. 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 analysiert zu werden.

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

Weitere Informationen siehe *Protokollierte Informationen (Entire Operations Log) anzeigen* im *Benutzerhandbuch*.

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.

Eine Liste aller für ein bestimmtes Datum geplanten Jobs 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 zur Verfügung.

Sie können mit dem Editor folgende Datentypen erstellen, anzeigen oder ändern:

- 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

5 Entire Operations-Funktionen

- 58
- Verwaltung der Job-Netzwerke 58
- Zeitplanung für Job-Netzwerke 58
- Job-Verwaltung 58
- Job-Zeitplanung 59
- Kalender-Definition 59
- Verwaltung der Bedingungen 60
- Job-Ende-Prüfung und -Aktionen 61
- Dynamische JCL-Generierung (MACRO-Funktionalität) 62
- Editor für System-Objekte 67
- Berichtsfunktionen 68
- Cross-Referenzen 69
- Benachrichtigung 70

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

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 Beschreibung 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 Endezeit 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 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.

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 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*.

Dynamische JCL-Generierung (MACRO-Funktionalität)

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 **Verwendbare Symboltabellen** bestimmt werden (siehe *In einem Netzwerk oder Job verwendbare Symboltabellen auflisten* 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 Betriebssystemen an (z/OS, 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 Scripts 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 im Menü **Allgemein** den Menüpunkt **Berichte** auswählt.

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.